A Comparison of the State-of-the-Art Reinforcement Learning Algorithms for Health-Aware Energy & Emissions Management in Zero-emission Ships

Reddy, Namireddy Praveen; Skjetne, Roger; Os, Oliver Stugard; Papageorgiou, Dimitrios

Published in:
IEEE Journal of Emerging and Selected Topics in Industrial Electronics

Link to article, DOI:
10.1109/JESTIE.2023.3331230

Publication date:
2024

Document Version
Peer reviewed version

Link back to DTU Orbit

Citation (APA):
Reddy, N. P., Skjetne, R., Os, O. S., & Papageorgiou, D. (2024). A Comparison of the State-of-the-Art Reinforcement Learning Algorithms for Health-Aware Energy & Emissions Management in Zero-emission Ships. *IEEE Journal of Emerging and Selected Topics in Industrial Electronics*, *5*(1), 149 - 166. https://doi.org/10.1109/JESTIE.2023.3331230

A Comparison of the State-of-the-Art Reinforcement Learning Algorithms for Health-Aware Energy & Emissions Management in Zero-emission Ships

Namireddy Praveen Reddy, *Student Member, IEEE,* Roger Skjetne, *Senior Member, IEEE,* Oliver Stugard Os, *Student Member, IEEE,* and Dimitrios Papageorgiou, *Member, IEEE*

***Abstract*—Zero-emission ships (ZES) have gained interest to comply with the stringent regulations of international maritime organization. One way to build ZES is the hybridization of fuel cells with batteries. Traditionally, for a newly built ship, the Energy & Emissions Management System (EEMS) is designed based on the initial condition of the fuel cells and batteries and used with fixed parameters in future execution. However, for a fuel cell and battery ZES, the EEMS gradually becomes sub-optimal since the characteristics of fuel cells and batteries are continuously changing due to aging and degradation. In this paper, a reinforcement learning (RL) based EEMS is developed such that it can learn and adapt continuously to changes in the fuel cell/battery characteristics. Within RL, different types of algorithms such as double deep Q learning (DDQL), soft actor-critic (SAC), and proximal policy optimization (PPO) are implemented. The results are benchmarked against those of a typical rule-based EEMS. Each RL algorithm is trained with four reward function formulations; negative cost (r_1), negative quadratic cost (r_2), inverse cost (r_3), and inverse quadratic cost (r_4). The results demonstrate that health-aware EEMS can minimize fuel consumption and component degradation costs. r_1 has led to the lowest operational expenses (OPEX) followed by r_2, while r_3 and r_4 have high OPEX. Among the three algorithms, the DDQL led to the lowest reward followed by the SAC and then the PPO, when trained with r_1 and r_2.**

***Index Terms*—Energy & emissions management strategy, Hybrid power system, Intelligent control, Reinforcement learning, Zero-emission ship.**

I. Introduction

ZERO-emission ships (ZES) have generated considerable research and development interest in recent years. A main driver for ship manufacturers and owners for investigating potentially low- or preferably zero-emission solutions, is to comply with the International Maritime Organization's (IMO) regulations for designated emission-controlled areas (ECAs) [1]. ZES is defined as a ship that does not produce emissions of greenhouse gases such as carbon dioxide (CO_2), methane (CH_4), and nitrous oxide (N_2O) according to Norwegian Maritime Authority (NMA) [2]. The schematics in Figure 1, adopted from [3], show a concept of ZES powered by fuel cells and batteries, where, $P_{FC\text{-ref}}$ is the fuel cell power reference, $P_{B\text{-ref}}$ is the battery power reference, V_{FC} is the fuel cell voltage, V_B is the battery voltage, V_{DC} is the DC bus voltage, $i_{FC\text{-ref}}$ is the fuel cell current reference, $i_{B\text{-ref}}$ is the battery current reference, i_{FC} is the fuel cell current, i_B is the battery current, u_{FC} is the control reference for the fuel cell DC/DC unidirectional converter, and u_B is the control reference for the battery DC/DC bidirectional converter. The control system of a ZES is abstracted into several autonomy layers, where the Energy and Emissions Management System (EEMS) will include algorithms for optimal guidance of the power plant in the middle layer.

Fuel cells and batteries have different features and operational challenges, which directly impact their lifetime and reliability, and their characteristics are continuously changing due to aging and degradation. Therefore, the loading of a fuel cell hybrid power system (FCHPS) should be optimized taking into account both the state of health and the operating expenses (OPEX). This is addressed in the executive control layer by the EEMS.

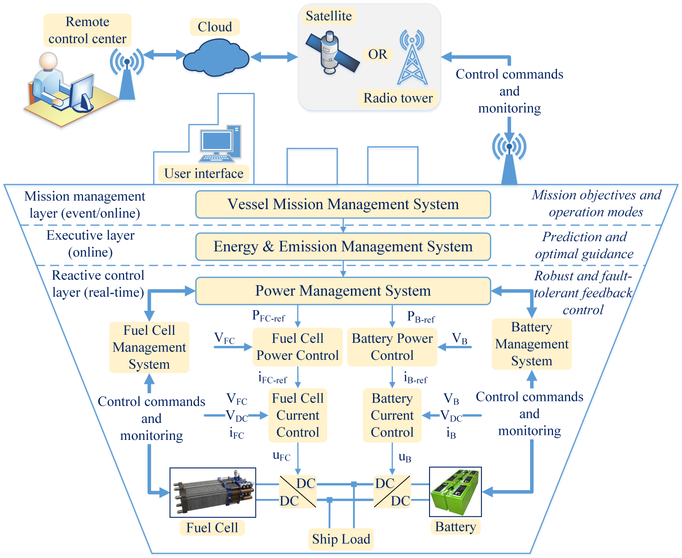

Remote control center
Cloud
Satellite
OR
Radio tower
Control commands and monitoring
User interface
Mission management layer (event/online)
Vessel Mission Management System
Mission objectives and operation modes
Executive layer (online)
Energy & Emission Management System
Prediction and optimal guidance
Reactive control layer (real-time)
Power Management System
Robust and fault-tolerant feedback control
$P_{FC\text{-ref}}$
$P_{B\text{-ref}}$
Fuel Cell Management System
V_{FC}
Fuel Cell Power Control
Battery Power Control
V_B
Battery Management System
$i_{FC\text{-ref}}$
$i_{B\text{-ref}}$
Control commands and monitoring
V_{FC} V_{DC} i_{FC}
Fuel Cell Current Control
Battery Current Control
V_B V_{DC} i_B
Control commands and monitoring
u_{FC}
u_B
DC DC
DC DC
Fuel Cell
Ship Load
Battery

Figure 1: A schematic of ZES [3].

Much of the existing literature considers minimizing fuel consumption as the main objective of the EEMS. However, the efficiency curve of the fuel cell tends to be flatter compared

Manuscript received 02 July 2023; revised 23 October 2023; accepted 05 November 2023. Date of publication 06 November 2023; date of current version 06 November 2023. (Corresponding author: Namireddy Praveen Reddy.)

Namireddy Praveen Reddy, Roger Skjetne, and Oliver Stugard Os are with the Department of Marine Technology, Norwegian University of Science and Technology, Trondheim, 7491 Norway, (e-mail: namireddy.p.reddy@ntnu.no; roger.skjetne@ntnu.no; oliverstugardos@gmail.com)

Dimitrios Papageorgiou is with the Department of Electrical Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark, e-mail: dimpa@dtu.dk

to that of the internal combustion engine, and as such, its operating efficiency is much less dependent on active decisions made by the EEMS. On the other hand, high cost and low lifetime are the main concerns for fuel cells, hindering their widespread commercialization [4]. The battery's lifetime is mainly dependent on how it is operated. Batteries in hybrid electric vehicles typically last several years, while fuel cells last several thousand hours of operation. This depends on many factors, e.g., how the fuel cell and battery are operated as discussed in [5], [6]. Hence, consideration of the lifetime of fuel cells and batteries must be included in the design of the EEMS. Many studies are reported on the development of EEMS methods for transport applications, such as [7], [8] on automotive applications and [9]–[12] on ship power systems (SPS).

The EEMS algorithms can be classified as rule-based and optimization-based methods [13], [8]. The rule-based strategies are easy to understand and implement but are typically not adaptive to prevailing sailing conditions and, thus, may give sub-optimal solutions [14]. Optimization-based energy & emissions management strategies are further divided into offline and online methods. Though there are many offline optimization methods, dynamic programming (DP) is widely used [8] since it provides global optimal solutions. However, the prerequisite for DP optimization is that the load profile should be known in advance, which implies large uncertainties in shipping applications. Also, dynamic programming suffers from the infamous 'Curse of Dimensionality' [15], [16]. Typical online optimization methods include equivalent consumption minimization strategy (ECMS) and model predictive control (MPC). The equivalence factor in ECMS requires tuning for real-time sailing conditions to find their optimal values in a trial-and-error manner, which makes ECMS hard to implement [9], [17]. The performance of MPC depends on the prediction accuracy and horizon length of the predicted sailing profile [18], [19].

Many of the existing EEMS methods are based on classical control methods, which require tedious system identification, construction of detailed mathematical models, and significant effort in developing control synthesis. In addition, classical control methods are not adaptive to real-time conditions and may give sub-optimal solutions for new operating profiles or system characteristics. In the quest to find a remedy for these problems, learning-based methods are gaining interest [20]. Within machine learning, reinforcement learning (RL) has attracted attention in recent years mainly because of its adaptability and model-free implementation. Though RL as a concept was proposed in the early 1980s, its potential to solve real world problems has been demonstrated through several pioneering works by Google's DeepMind [21]–[25]. The RL-based controller continuously optimizes the control policy for the evolving system and sailing conditions. Many advantages of RL can be realized in applications where the system model is unknown, not accurate, or continuously changing. A relevant application is FCHPS, where the characteristics of battery packs keep changing as the battery packs go through numerous charging/discharging cycles [26], the characteristics of a fuel cell stack keep changing due to aging and degradation [6], and shiploads are uncertain and difficult to predict as they depend on many random parameters such as weather conditions, ocean currents, and other external factors. These system uncertainties can be mitigated by an RL-based controller that may continuously learn the optimal control policy for the changing component characteristics and uncertain environment and loads [27].

Conventional RL-based energy management was proposed in [28] for optimizing the fuel economy of a hybrid electric tracked vehicle; the authors compared the fuel consumption obtained from RL with rule-based and stochastic dynamic programming strategies to prove the optimality. Conventional RL was also employed for online energy management to minimize the total energy loss of the hybrid energy storage system in a plug-in hybrid electric vehicle [29]; the results were compared with a rule-based strategy and showed that RL could lessen the total energy loss and improve the system efficiency under varying conditions. Going one step further, the authors in [30] used two novel velocity predictors together with conventional RL for predictive energy management of a parallel HEV, and the results of predictive energy management were compared with non-predictive and dynamic programming to validate the optimality. To overcome the high computational requirements of conventional RL, a deep reinforcement learning framework was implemented by the authors in [16] for energy management to optimize the fuel economy in a hybrid electric bus by incorporating simulated terrain information. Deep reinforcement learning was also implemented by the authors in [31] for energy management with the aim of minimizing fuel consumption in a hybrid electric bus by incorporating traffic information. An energy management strategy, based on the double deep Q-learning (DDQL) algorithm, was proposed by the authors in [32] for optimizing fuel consumption of a hybrid electric tracked vehicle; the results showed that DDQL has better performance than conventional deep reinforcement learning in terms of convergence during the training process and also in optimizing the fuel consumption. Khalatbarisoltani *et al*. [33] integrated model predictive control with federated reinforcement learning for decentralized energy management of fuel cell vehicles; the proposed method performs better than the centralized and fixed-horizon MPC approaches in terms of its precision, convergence speed, and scalability.

Among many RL algorithms proposed in the existing literature, tabular Q-learning (TQL) and deep Q-learning (DQL) algorithms are some of the most popular RL algorithms. TQL suffers from the "curse of dimensionality" and is feasible for only low-dimension state and action spaces [27]. The difference between TQL and DQL is the deep neural network (DNN) that approximates the Q-value. Traditional DQL has several issues such as overoptimism and instability during training, both caused by the fact that the same DNN is used for the selection and evaluation of control action [21]. To overcome these issues, Hasselt *et al*. [24] proposed the DDQL. Additionally, many state-of-the-art RL algorithms such as the soft actor-critic (SAC) by [34] and the proximal policy optimization (PPO) by [35], are proposed recently. Although recent research has primarily explored reinforcement learning (RL) methods in transportation applications, a comprehensive comparison of state-of-the-art RL algorithms, such as DDQL, SAC, and PPO, particularly within the context of marine transport is missing, which highlights a significant research gap. Implementing these RL algorithms in real-life scenarios with continuous state-action spaces is computationally intensive and complex, necessitating extensive iterations for training and hyperparameter tuning. This calls for the development of computationally efficient yet accurate models for fuel cells and batteries that can effectively capture aging and degradation effects—an area of active research. Furthermore, the performance of reinforcement learning algorithms primarily depends on the formulation of the reward functions. To the best of the authors' knowledge, investigations of different ways of formulating reward functions using the realistic operational cost function have been missing in the literature.

The contributions of this paper, which arise from the quest to bridge these critical research gaps in the existing literature, can be summarised as:

1) Implementation of RL algorithms, including DDQL, SAC, and PPO, to develop an EEMS that continuously adapts its policies to accommodate the evolving characteristics of fuel cells and batteries, considering the impact of degradation and uncertain shiploads. This contribution extends the applicability of RL methods to the maritime transportation domain, offering dynamic and efficient solutions to energy management challenges.
2) To facilitate the computationally efficient training of RL algorithms, a novel hybrid model is proposed. This model combines linearized polarization curve models whose parameters dynamically adjust in response to nonlinear aging and degradation effects. However, the performance of trained RL agents is validated with nonlinear models combined with nonlinear aging and degradation effects. This approach represents a step in the ease of implementing RL methods in maritime transportation.
3) A realistic cost function is formulated to represent the operational expenses of Fuel Cell and Battery Hybrid Power Systems (FCHPS), including a unique formulation for battery degradation cost tailored to maritime transport applications. The paper also explores and experiments with four alternatives for formulating the reward function, focusing on operational cost representation. This contribution ensures a comprehensive understanding of the cost implications associated with various operational states, enhancing the realism of RL algorithms in maritime transportation scenarios.

These contributions advance the implementation of state-of-the-art RL algorithms in addressing energy management challenges, offering practical solutions and insights. By addressing critical gaps in RL algorithm implementation, models for energy system components, and cost function formulation, this research possesses implications for both practical applications and theoretical understanding.

II. Reward function formulation (RFF)

The performance of RL algorithms strongly depends on how the reward function is formulated. In this work, the reward is formulated based on the total operational expenses (OPEX) including the costs of fuel and component degradation. In this section, the overall goal of minimizing the cost of OPEX of the FCHPS denoted as C_{opex}, is covered. C_{opex} is given by

$$C_{opex} = C_{fuel} + C_{FC,deg} + C_{bat,deg}, \tag{1}$$

where $C_{fuel} \geq 0$ is the cost of fuel, $C_{FC,deg} \geq 0$ and $C_{bat,deg} \geq 0$ are the costs due to degradation of the fuel cell (FC) and battery, respectively. These costs are elaborated in the subsections below; C_{fuel} and $C_{bat,deg}$ are formulations proposed in this paper while $C_{FC,deg}$ is based on Fletcher *et al.* [4]. There are several ways to formulate the reward function using the OPEX (1); four different ways will be explored: r_1 is the negative cost (NC),

$$r_1 = -C_{opex} \tag{2}$$

r_2 is the negative quadratic cost (NQC),

$$r_2 = -C_{opex}^2 \tag{3}$$

r_3 is the inverse cost (IC),

$$r_3 = \frac{1}{C_{opex} + \varepsilon} \tag{4}$$

r_4 is the inverse quadratic cost (IQC),

$$r_4 = \frac{1}{C_{opex}^2 + \varepsilon} \tag{5}$$

where $0 < \varepsilon << 1$ is added to the denominator to ensure that errors are not encountered when C_{opex} approaches zero during the simulation.

A. Fuel cost

A fuel cell uses hydrogen as fuel to generate power to supply the power loads and to charge the battery. The calculation of fuel cost consists of three components: the fuel consumed by the FC ($C_{FC,fuel}$) [9], the equivalent consumption to supply the battery power losses ($C_{bat,loss}$), and the equivalent consumption to supply the change in SOC of the battery ($C_{\Delta soc}$):

$$C_{fuel} = C_{FC,fuel} + C_{bat,loss} + C_{\Delta soc} \tag{6}$$

$$C_{FC,fuel} = C_{H_2} \underbrace{\tfrac{N}{F} \tfrac{M_{H_2}}{1000} I_{FC} \mathrm{d}t}_{H_{2_{cons}}} \tag{7}$$

$$C_{bat,loss} = C_{H_2} \frac{N}{F} \frac{M_{H_2}}{1000} \frac{R_{bat} I_{bat}^2}{V_{FC}} \; \mathrm{d}t \tag{8}$$

$$C_{\Delta soc} = C_{H_2} \frac{N}{F} \frac{M_{H_2}}{1000} \frac{(SOC_{init} - SOC_{end}) Q_{nom}}{V_{FC}} \tag{9}$$

Here, C_{H_2} is the price of fuel per kg in [\$/kg], set to 6 \$/kg [36], $H_{2_{cons}}$ is the total consumed hydrogen mass in [kg], N is the number of cells in the stack, F is the Faraday constant in [C/mol], M_{H_2} is the molar mass of hydrogen in [g/mol], I_{FC} is the FC current in [A], dt is the evaluation time step, R_{bat} is the battery internal resistance in [Ω], I_{bat} is the battery current in [A], V_{FC} is the fuel cell voltage in [V], SOC_{init} and SOC_{end} are the initial and final states of charge of the battery, respectively, expressed as a fraction of nominal capacity, and Q_{nom} is the nominal battery capacity in [Ws].

B. Fuel cell cost

A major portion of the operating cost in FCHPS is due to the degradation of the fuel cell. Computing the degradation cost precisely using a mathematical model of multiple complex chemical phenomena is computationally inefficient. Therefore, avoiding a high-fidelity model, a simplified approach based on [4] is applied to calculate the degradation cost. The EMS can optimize the degradation by taking the following operational actions as explained in [4]:

- **FC Low power operation (FC-LPO)**: Minimize running the fuel cell at low current (power) to limit reduction of the catalyst layer due to the formation of oxides. Operation of the fuel cell at lower power than a lower limit of approximately 10 % of the rated capacity contributes to degradation. Hence, the cost factor D_{low} is defined by

$$D_{low} := \begin{cases} \alpha_{low} \frac{0.1P_{max} - P_{FC}}{0.1P_{max}} \, \mathrm{d}t, & \text{if } P_{FC} < 0.1P_{max} \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

  where α_{low} is the degradation rate for the low-power operation condition.
- **FC High power operation (FC-HPO)**: Minimize running the fuel cell at a high current (power) to prevent reactant starvation that can lead to reduction of the catalyst layer. Moreover, the excessive temperatures due to high current can lead to the damage of cathode support and degradation of the fuel cell membrane. Operating the fuel cell at higher power than an upper limit of 90 % of the rated capacity contributes to degradation. Correspondingly, the cost factor D_{high} is defined as

$$D_{high} := \begin{cases} \alpha_{high} \frac{P_{FC} - 0.9P_{max}}{0.1P_{max}} \, \mathrm{d}t, & \text{if } P_{FC} > 0.9P_{max} \\ 0, & \text{otherwise}, \end{cases} \tag{11}$$

  where α_{high} is the degradation rate for the high power operation condition.
- **Fuel cell transients (FC-T)**: Limit the rate of change of the fuel cell power and minimize the transient loads to maintain a stable temperature and humidity in the cell. This also prevents local fuel starvation [37]. In this work, the maximum rate of change in fuel cell power is limited to 10 % of the rated capacity per second based on the work by Zhang *et al.* [38]. The degradation cost factor due to high power transients D_{trans} is correspondingly defined by

$$D_{trans} = \beta \left| \frac{\mathrm{d}P_{FC}}{\mathrm{d}t} \right| \mathrm{d}t = \beta \left| \mathrm{d}P_{FC} \right|, \tag{12}$$

  where β is the degradation rate due to high power transients.
- **Startup/shutdown cycles**: Minimize the number of start-up/shut-down cycles to prevent the nonuniform distribution of fuel and thereby prevent localized starvation. The FC is assumed to be running all the time as the aim is to minimize OPEX during a sailing trip, thus avoiding the cost of startup/shutdown. The degradation cost factor due to start-up/shut-down cycles D_{cycles} is thus neglected, that is,

$$D_{cycles} = 0. \tag{13}$$

The total fuel cell degradation cost $C_{FC,deg}$ becomes

$$C_{FC,deg} = C_{FC} \cdot (D_{low} + D_{high} + D_{trans} + D_{cycles}), \tag{14}$$

where C_{FC} is the acquisition cost of the fuel cell. The degradation parameters α_{low}, α_{high}, and β are summarized in Table I [4]; these values are based on laboratory experiments [39], [40]. α_{low} and α_{high} were applied as constant values in [4]. To make the cost more realistic, α_{low} and α_{high} are applied as linearly varying variables while keeping the average value equal to the desired constant value. The degradation rates and costs are scaled to suit the size of the fuel cell stack. The nominal FC voltage is 629 V, the fuel cell has 900 cells, the end of life (EOL) is considered when the open circuit voltage degrades to 10 % of the rated voltage, and the FC acquisition cost per kW is set to 75 $ [41].

Table I: FC degradation rates and costs [4].

Parameter	**Operating condition**	**Deg. rate**	**Deg. cost**
α_{low}	High power	20.34 μV/h	1.57 $/h
α_{high}	Low power	23.48 μV/h	1.81 $/h
β	Transient loading	0.0441 μV/kw	0.0034 $/kW

C. Battery cost

Similar to the fuel cell's degradation, the battery's degradation is a complex electro-chemical phenomenon. Though there are many factors that influence battery aging and degradation as shown in Figure 2 adopted from [42], the two main factors are: 1) cyclic aging, and 2) calendar aging [5]. Calendar aging is due to inherent battery aging and mainly depends on time, therefore, it is not considered in this work. The main parameters determining the cyclic aging of a battery are the state-of-charge (SOC), the depth-of-discharge (DOD), and the C-rate; see Figure 2 for details.

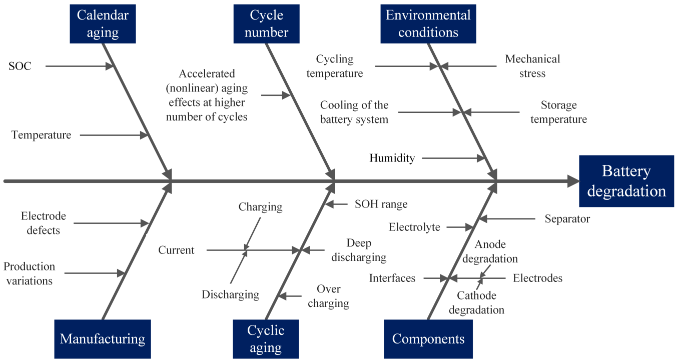


Figure 2: Ishikawa diagram of the various aging factors leading to Lithium-ion battery degradation [42].

- **State-of-charge**: Typically, the battery SOC is constrained within $SOC \in [20\%, 80\%]$ to prolong the battery lifetime. The upper limit $SOC_{max} = 80\%$ is necessary to avoid overcharging and the lower limit $SOC_{min} = 20\%$ is necessary to avoid deep discharging; both overcharging and deep-discharging cause significantly higher degradation. The cost factor D_{SOC} is defined as

$$D_{SOC} := \gamma \cdot \left|SOC_{ref} - SOC(t)\right| \, \mathrm{d}t, \tag{15}$$

where $\gamma \in [0, \infty)$ is the degradation rate accounting for the SOC limits (overcharging and deep discharging) and $SOC_{ref} = 50\%$.

- **Depth-of-discharge**: The DOD consists of two parameters, DOD_{charge} and $DOD_{discharge}$, which define how much the battery has charged or discharged without interruption. Koller *et al.* [43] suggested a model for battery degradation due to DOD. Xu *et al.* [44] argued that the models used in the literature did not give an adequate representation of battery degradation. Wang *et al.* [45] performed tests on $LiFePO_4$ battery under different operating conditions such as temperature, DOD, and C-rate in order to find a function that estimated the battery degradation. They also conducted experiments on cyclic aging until the EOL was reached, and then switched the degradation function to Ah-throughput (Ah_{th}). After several experiments, the capacity loss estimate was proposed in [45] is

$$Q_{loss} = A\exp\left[\frac{-E_a + BC_{rate}}{RT}\right](Ah_{th})^z, \tag{16}$$

where A is a pre-exponential factor, E_a is the activation energy of the $LiFePO_4$ battery examined, B is an exponential factor weighting the C-rate properly, z is a factor to emphasize the effect of the Ah-throughput, R is the gas constant, and T is the battery cell temperature.
- **C-rate**: In an attempt to quantify the effect from DOD and C-rate on battery degradation, Chen *et al.* [46] used (16) to model the capacity loss in the battery as a function of DOD and C-rate. First, Ah_{th} could be calculated by

$$Ah_{th} = Q_{nom} \cdot DOD \cdot N \tag{17}$$

where N is the number of cycles. Using (17), the number of cycles the battery can sustain before its EOL, with a given DOD and C-rate, is quantified by rearranging and combining (16) and (17).

$$N = \left[\frac{Q_{loss}}{Ae^{\left(\frac{-E_a+BC_{rate}}{RT}\right)}}\right]^{\frac{1}{z}} \frac{1}{Q_{nom} \cdot DOD}, \tag{18}$$

where Q_{loss} is the battery capacity loss allowed before its EOL and Q_{nom} is the nominal capacity of battery. To get to (18), [46] assumed a constant C-rate; however, it would rarely be the case for a real operation where a varying power demand will cause the C-rate to fluctuate. Thus, the average C-rate during a given charge/discharge cycle is used in the above equation. Furthermore, the pre-exponential factor A varies to some degree with different C-rates. So, A is set equal to the C-rate corresponding to 2 C, which is considered the nominal C-rate for the battery in marine transport applications. The values of the parameters used in Equation (18) are based on [45], [46] and given in Table II. Figure 3 is based on Eqn. 18 and shows that the number of cycles (N) decreases with the increase in C-rate as well as an increase in DOD. The cost of half a charge/discharge cycle, given an average C-rate, is then calculated by dividing by 2N, that is,

$$D_{DOD,Crate} := \frac{1}{2N}. \tag{19}$$

The total cost of running the battery becomes

$$C_{bat,deg} = C_{bat} \cdot (D_{SOC} + D_{DOD,Crate}), \tag{20}$$

where C_{bat} is the price of the lithium-ion battery.

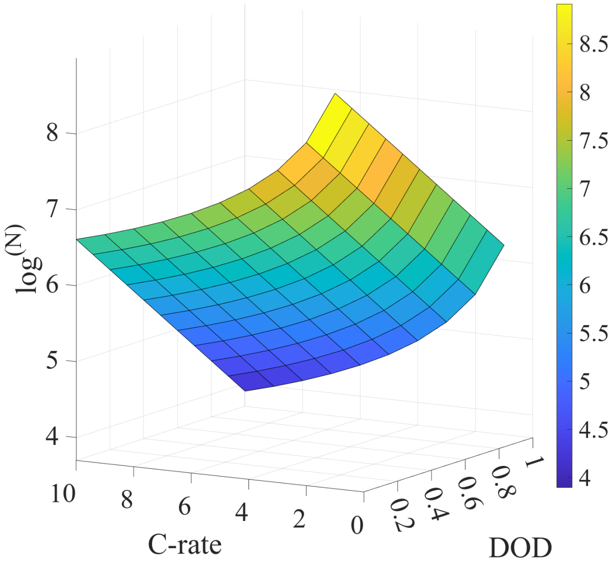


Figure 3: Effect of DOD and C-rate on the number of charging/discharging cycles.

Table II: Li-ion battery parameters [45], [46].

Parameter	Description	Value
Q_{loss}	Maximum allowed capacity loss	20 %
E_a	Activation energy	31.500 J/mol
A	Pre-exponential factor	19.300 kWh
B	Exponential effect of C_rate	370.3 J/(mol.A)
z	Power law factor	0.55
R	The gas constant	8.314 J/(K · mol)
T	Battery cell temperature	298.15 K

III. Ship Power System Model

The FCHPS has several components for power generation, distribution, and power consumption that can be mathematically represented by simple to complex models depending on the intended use. One of the objectives of this work is to train the RL algorithms to minimize fuel consumption and degradation of components, which requires generating a sufficient amount of data. For this purpose, a simple model of an FCHPS that includes degradation effects is necessary, whose modeling aspects are covered in this section. In this work, reduced models of sufficient fidelity of the fuel cell and battery are proposed. In the proposed setup, linearized polarization curve models combined with nonlinear aging effects are used for training the RL agents, whereas nonlinear models are used for validating the performance of the trained RL agents.

In this study, a passenger ferry, which typically encounters propulsion loads of around 100 kW during cruising powered by a 120 kW fuel cell and a 50 kWh Li-ion battery is chosen. The component selection and sizing are based on the existing literature works of Bassam *et al.* [47] and Sulaiman *et al.* [8] among others in similar applications. It's worth noting that while there is room for optimizing component sizing, this particular aspect is not the primary focus of this research.

A. Fuel cell model

1) Nonlinear model

A generic nonlinear fuel cell model was proposed in [48], based on the manufacturer's data sheet; the proposed model is shown in Figure 4, which depicts a fuel cell stack as a controlled voltage source (E) in series with an internal resistance (R_{ohm}). The output FC power is calculated by

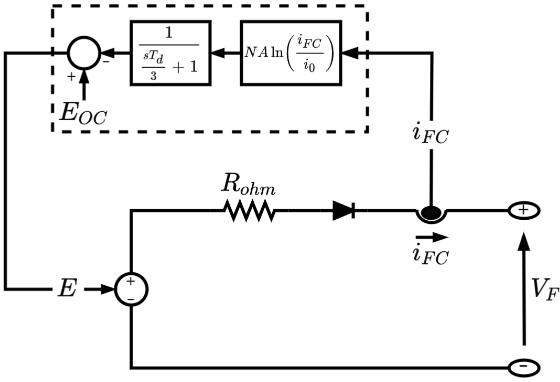

$\frac{1}{\frac{sT_d}{3}+1}$
$NA\ln\left(\frac{i_{FC}}{i_0}\right)$
+
-
E_{OC}
i_{FC}
R_{ohm}
i_{FC}
E
+
-
V_{FC}
+
-

Figure 4: A generic fuel cell model [48].

$$P_{FC} = i_{FC}V_{FC}. \tag{21}$$

The controlled voltage source of the fuel cell is [48], [49]

$$E = E_{OC} - \overbrace{N_{FC}A_{FC}\ln\left(\frac{i_{FC}}{i_0}\right)}^{\text{Activation loss}}, \tag{22}$$

where E_{OC} is the open circuit voltage (OCV), N_{FC} is the number of cells, A_{FC} is the Tafel slope, i_{FC} is the FC current, and i_0 is the exchange current. The OCV is obtained by the Nernst equation, which is affected by the temperature, partial pressures of hydrogen and air, as well as their concentrations. The remaining values are found in [48] as follows:

$$N_{FC}A_{FC} = \frac{(V_1 - V_{max})(i_{max} - 1) - (V_1 - V_{min})(i_{min} - 1)}{\ln(i_{min})(i_{max} - 1) - \ln(i_{max})(i_{min} - 1)} \tag{23}$$

$$R_{ohm} = \frac{V_1 - V_{max} - N_{FC}A_{FC}\ln(i_{min})}{i_{min} - 1} \tag{24}$$

$$i_0 = \exp\left(\frac{V_1 - E_{OC} + R_{ohm}}{NA}\right). \tag{25}$$

R_{ohm} is the internal fuel cell resistance, V_1 is the output voltage at 1 A, V_{nom} and i_{nom} are the voltage and the current at the nominal operation point, respectively and V_{min} and i_{max} are the voltage and the current at the maximum power point, respectively. The polarization curve in Figure 5 shows how the output voltage varies with the current and describes the fuel cell characteristics. The values can be found by examining four points on the polarization curve as described by Motapon *et al.* [48]. The FC output voltage, V_{FC}, can then be calculated as

$$V_{FC} = E - \overbrace{R_{ohm}i_{FC}}^{\text{Ohmic loss}}. \tag{26}$$

Figure 5 shows a generic nonlinear polarization curve, which consists of the activation, ohmic, and mass transport regions. The output voltage varies nonlinearly in activation and mass transport regions, whereas it varies approximately linearly in the ohmic region. The regional differences come from internal losses that originate from activation losses, ohmic losses, and concentration losses.

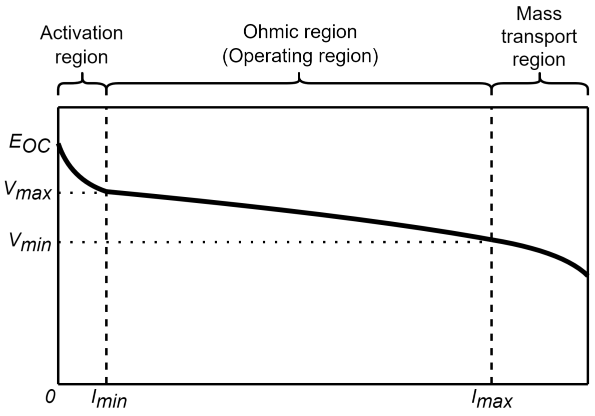


Figure 5: Nonlinear polarization curve of the fuel cell.

2) Linear model

A simplified version is obtained by piecewise linearization of the generic polarization curve, assuming that the fuel cell does not operate in the activation region and mass transport region (see Figure 6). The consequence of this assumption is that the fuel cell current is constrained by a minimum value (I_{min}) and a maximum value (I_{max}). This is a reasonable assumption, as the efficiency drops drastically and degradation rates are very high in the activation and mass transport regions. The linearization parameters are

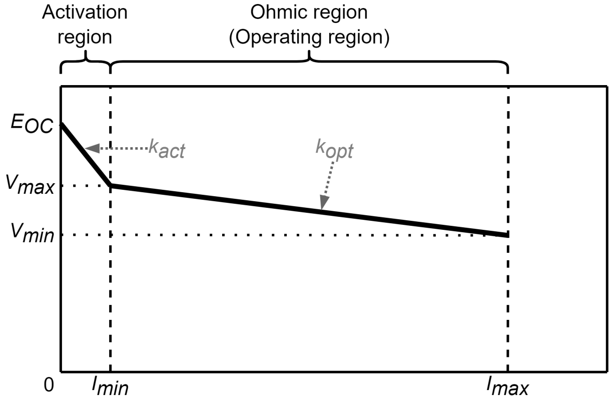


Figure 6: Linear approximation of fuel cell's polarization curve.

$$k_{act} := \frac{E_{OC} - V_{max}}{I_{min}} \tag{27}$$

$$k_{opt} := \frac{V_{max} - V_{min}}{I_{max} - I_{min}}, \tag{28}$$

and the voltage and power becomes

$$V_{FC} = \begin{cases} V_{max} - k_{opt}(i_{FC} - I_{min}), & \text{if} \quad i_{FC} > I_{min} \\ E_{OC} - k_{act} i_{FC}, & \text{otherwise} \end{cases} \tag{29}$$

$$P_{FC} = i_{FC} V_{FC}. \tag{30}$$

The parameters used for the linearized polarization curve based on [48] are given in Table III.

Table III: Fuel cell parameters [48].

Parameter	Description	Value
E_{OC}	Open circuit voltage	900 V
V_{max}	Voltage at start of Ohmic region	800 V
V_{min}	Voltage at end of Ohmic region	430 V
I_{min}	Current at start of Ohmic region	20 A
I_{max}	Current at end of Ohmic region	280 A

To model the internal FC delay, the rate of change in current is constrained to 10 % of I_{max} per second. This constraint encapsulates the limitations in the dynamic capabilities of the fuel cell. The fuel supply system has slow dynamics due to the mechanical valves, which cause fuel starvation during high transients in the fuel cell current. The consequence is accelerated degradation of the fuel cell. Therefore, the rate of change in the current should be limited so that the model accounts for the slow dynamics of the fuel cell, that is,

$$|\Delta I_{FC}| \leq 0.1 \ I_{FC,max} \mathrm{d}t. \tag{31}$$

3) Aging effects

Fuel cell degradation is reflected physically in the fuel cell voltage. A 10 % loss of fuel cell voltage under rated current is considered the end of life (EOL) [40]. Computing the fuel cell voltage degradation using physics-based models is cumbersome. Therefore, empirical models developed based on experimental data in [39], [40] are used in this work. The fuel cell voltage degradation ($V_{FC,deg}$) is computed by

$$V_{FC,deg} = D_{low} + D_{high} + D_{transients} + D_{cycles} \tag{32}$$

$$E_{OC}(i+1) = E_{OC}(i) - V_{FC,deg}. \tag{33}$$

B. Battery Model

1) Nonlinear model

A generic battery dynamic model based on [50], [51] is shown in Figure 7. For the lithium-ion battery type, the model uses the following equations.

Discharge model ($i^* > 0$):

$$f(it, i^*, i) = E_0 - K\frac{Q_{nom}}{Q_{nom} - it} i^* - K\frac{Q_{nom}}{Q_{nom} - it} it + Ae^{-Bit} \tag{34}$$

Charge model ($i^* < 0$):

$$f(it, i^*, i) = E_0 - K\frac{Q_{nom}}{it + 0.1Q_{nom}} i^* - K\frac{Q_{nom}}{Q_{nom} - it} it + Ae^{-Bit} \tag{35}$$

where battery voltage E is obtained by the function $f(it, i^*, i)$, i is the battery current, i^* is the low-frequency battery current dynamics, it is the extracted capacity, E_0 is battery OCV, Q_{nom} is the nominal battery capacity, K is the polarization constant, A is the exponential voltage, and B is the exponential capacity.

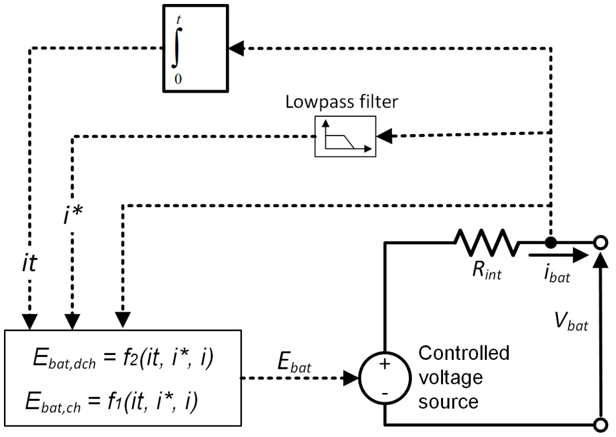

$\int_0^t$
Lowpass filter
i^*
it
R_{int}
i_{bat}
V_{bat}
$E_{bat,dch}$ = $f_2(it, i^*, i)$
$E_{bat,ch}$ = $f_1(it, i^*, i)$
E_{bat}
+
-
Controlled voltage source

Figure 7: A generic model of Li-ion battery [50].

A Li-ion battery's characteristics can be represented using a polarization curve similar to that of a fuel cell. The nonlinear polarization curve is shown in Figure 8 which is based on the dynamic battery model developed by Tremblay *et al.* [50]. The polarization curve is visualized by plotting the output voltage as a function of capacity, which shows how the output voltage varies with capacity or SOC. In the exponential zone, the battery voltage varies exponentially with the SOC. The battery is usually operated with predetermined SOC limits, typically $Q_{min} = 20\%$, $Q_{max} = 80\%$. In the operating zone, the voltage is varied approximately in a linear manner. This is where the battery thrives and has the best operating range to prolong its life. When the SOC falls below Q_{min}, the battery voltage decreases exponentially.

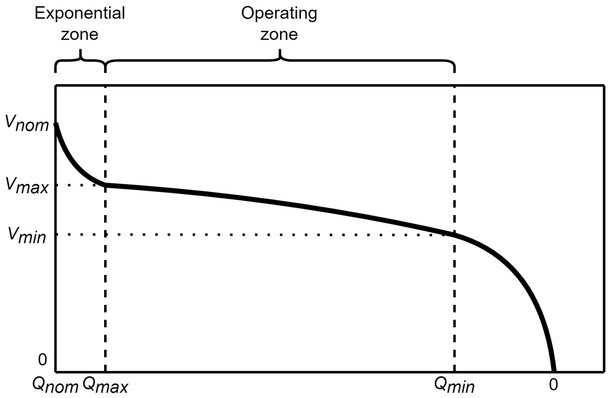


Figure 8: Nonlinear polarization curve of Li-ion battery.

2) Linear model

The polarization curve can be linearized as shown in Figure 9 with the assumption that the battery SOC is constrained within $SOC \in [20\%, 80\%]$. This is considered fair, as batteries should not be operated at very high or very low SOC due to unwanted battery degradation. The relationship between the SOC and the open circuit voltage, E_{OC}, is found from the linearized battery characteristic curve.

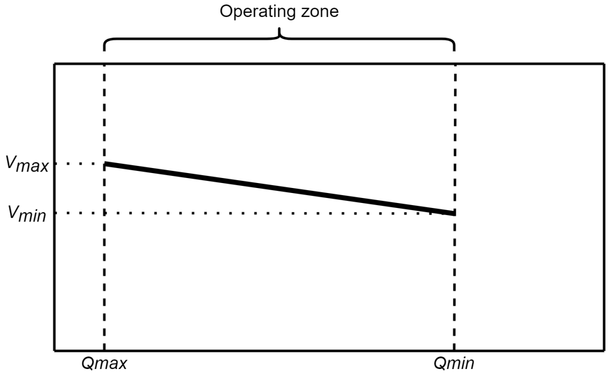


Figure 9: Linearized polarization curve of Li-ion battery.

$$E_{OC} = V_{max} - \frac{V_{max} - V_{min}}{Q_{max} - Q_{min}}(Q - Q_{min}) \tag{36}$$

The battery output voltage is calculated by compensating for the internal resistance, that is

$$V_{bat} = E_{OC} - R_{int} i_{bat}. \tag{37}$$

and the output power is

$$P_{bat} = V_{bat} i_{bat}. \tag{38}$$

The variation in SOC is based on a simple calculation of how much current enters or leaves the battery. The resulting model for change in capacity Q is

$$Q(t) = Q(0) - \int_0^t i_{bat} \mathrm{d}t. \tag{39}$$

The capacity Q of the battery is given in Ah. As a result, the SOC is updated in the following way at each time step of model simulation,

$$SOC(t + \mathrm{d}t) = SOC(t) - \frac{i(t)\mathrm{d}t}{3600 Q_{nom}}. \tag{40}$$

The relevant parameters for the battery model based on [50], [51] used in training RL algorithms are given in Table IV.

Table IV: Battery parameters [50], [51].

Parameter	Description	Value
Q_{nom}	Nominal capacity	50 kWh
Q_{max}	Maximal battery capacity	40 kWh
Q_{min}	Minimal battery capacity	10 kWh
V_{max}	Voltage at end of exponential zone	545 V
V_{min}	Voltage at end of nominal zone	430 V
$SOC(0)$	Initial SOC level	50 %

3) Aging effects

Battery degradation is reflected physically in battery capacity and internal resistance. Typically a 20% loss of battery capacity is considered the end of life. Battery capacity degradation Q_{bat} is computed based on [45] [46], according to

$$Q_{loss} = Q_{BOL} - Q_{EOL} \tag{41}$$

$$Q_{deg} = \frac{Q_{loss}}{N} \tag{42}$$

$$Q_{bat}(i+1) = Q_{bat}(i) - Q_{deg}, \tag{43}$$

where Q_{BOL} and Q_{EOL} are the battery capacities at the beginning and end of life, respectively.

C. Other components

1) Fuel cell converter

A unidirectional boost DC-DC converter interfaces the fuel cell stack with the DC bus. A simple model of a fuel cell converter from [9] is used with the assumption that the primary control (of the power electronic converter) works as intended. This assumption is reasonable as the goal is to use the model in the high-level controller, i.e., the EEMS. The model is

$$P_{FC,bus} = P_{FC} \eta_{FC,conv} \tag{44}$$

$$I_{FC,bus} = I_{FC} k \eta_{FC,conv} = \frac{P_{FC,bus}}{V_{DC}} \tag{45}$$

$$k := \frac{V_{FC}}{V_{DC}}, \tag{46}$$

where $P_{FC,bus}$ and $I_{FC,bus}$ are the FC power and FC current at DC bus, respectively, $\eta_{FC,conv}$ is the efficiency of fuel cell converter, and V_{DC} is the DC bus voltage.

2) Battery converter

A bi-directional buck-boost converter is used to interface the battery with the DC bus. A simple model from [9] is used to represent the battery converter

$$P_{bat,bus} = \begin{cases} P_{bat} \eta_{bb,dch} & \text{Discharging mode} \\ \frac{P_{bat}}{\eta_{bb,ch}} & \text{Charging mode} \end{cases} \tag{47}$$

$$I_{bat,bus} = \begin{cases} I_{bat} k \eta_{bb,dch} & \text{Discharging mode} \\ \frac{I_{bat} k}{\eta_{bb,ch}} & \text{Charging mode,} \end{cases} \tag{48}$$

where $k := \frac{V_{bat}}{V_{DC}}$, $P_{bat,bus}$ and $I_{bat,bus}$ are the battery power and current at the DC bus, respectively. $\eta_{bb,ch}$ and $\eta_{bb,dch}$ are the efficiencies of the buck-boost converter during charging and discharging, respectively.

3) *Shiploads*

The shiploads are represented by the load power profile of a typical passenger ferry based on [52], which is shown in Figure 10. The modes of operation in a typical passenger ship load profile can be classified as undocking, sailing (crossing), harbor maneuvering, and docking.

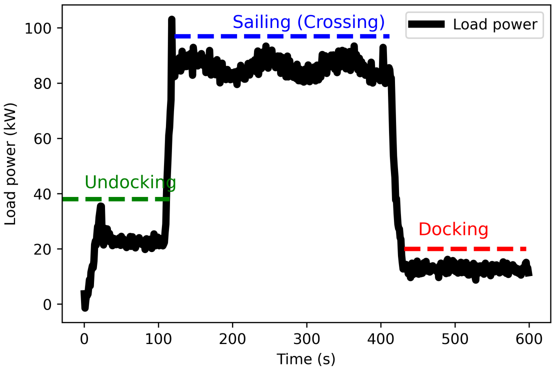


Figure 10: Load power profile of a typical passenger ferry [52].

IV. RL-based Energy & Emissions Management Strategies

In this section, three state-of-the-art RL algorithms implemented for EEMS: the DDQL, the SAC, and the PPO are elaborated. The aim of RL is to learn optimal policies by continuously interacting with a potentially uncertain environment while maximizing cumulative reward in sequential decision problems. Figure 11 shows the workflow of a typical RL-based EEMS in an SPS. For a given current state (s_k), the RL-based EEMS chooses the control action (a_k) based on a policy that seeks optimality. When a_k is executed, the SPS transitions from the current state s_k to the next state s_{k+1}. Based on s_k, a_k, and s_{k+1}, the reward (r_k) is computed, which is used to update the policy of the RL-based EEMS. Table V shows a qualitative comparison of the DDQL, the SAC, and the PPO methods, which are implemented in this work. The action space in the DDQL is discrete, which limits the action-space dimension. Both SAC and PPO can work on a continuous state-action space, which makes them feasible for real-world applications such as the FCHPS. SAC and PPO use the advantage factor as the main operator.

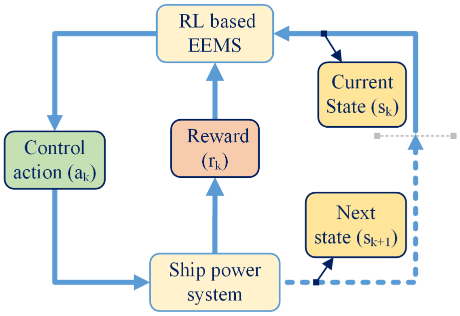


Figure 11: Generic RL algorithm.

Table V: Comparison of RL algorithms.

Algorithm	Policy	Action-space	State-space	Operator
DDQL	Off-policy	Discrete	Continuous	Q-value
SAC	Off-policy	Continuous	Continuous	Advantage
PPO	On-policy	Continuous	Continuous	Advantage

A. *Double Deep Q Learning (DDQL) Algorithm*

The schematic for the DDQL algorithm proposed by Hasselt *et al.* [24], can be observed in Figure 12, accompanied by the corresponding pseudo code available in algorithm 1. The main idea of the DDQL algorithm 1 is to reduce the issue of over-optimism in the DQL by using two identical DNNs denoted as critic network and target critic network for action selection and action evaluation, respectively. As shown in Figure 12, the critic network is trained at every iteration, using transitions (s_k, a_k, r_k, s_{k+1}) stored in experience buffer [21]. Whereas the target critic network copies the parameters of the critic network periodically. As the aim of the critic network is to predict the Q-value, the Bellman temporal difference (TD) error

$$TD_{error} = \underbrace{r_k + \gamma \max_a \hat{Q}(s_{k+1}, a, \hat{\theta})}_{\text{target Q value}} - \underbrace{Q(s_k, a_k, \theta)}_{\text{predicted Q value}} , \tag{49}$$

should be minimized. Here, γ is the discount factor, θ represents the parameters of the critic network, $\hat{\theta}$ represents the parameters of the target critic network, $\hat{Q}(s_{k+1}, a, \hat{\theta})$ is the maximum target Q value that can be obtained in the state s_{k+1}, and $Q(s_k, a_k, \theta)$ is the Q value for s_k and a_k. The loss function that the critic network aims to minimize is the square of the TD error, that is,

$$L_k(\theta) = [TD_{error}]^2 . \tag{50}$$

Minimizing $L_k(\theta)$ is achieved by performing gradient descent with respect to the parameters θ of the critic network. The gradient of the loss with respect to the parameters θ becomes $\nabla_\theta L_k(\theta) = -2TD_{error}\nabla_\theta Q(s_k, a_k, \theta)$, giving the update rule of the critic network

$$\begin{aligned} \theta_{k+1} &= \theta_k + \Delta\theta \\ \Delta\theta &= \alpha\,(TD_{error})\,\nabla_\theta Q(s_k, a_k, \theta)) \end{aligned}$$

where α is the learning rate.

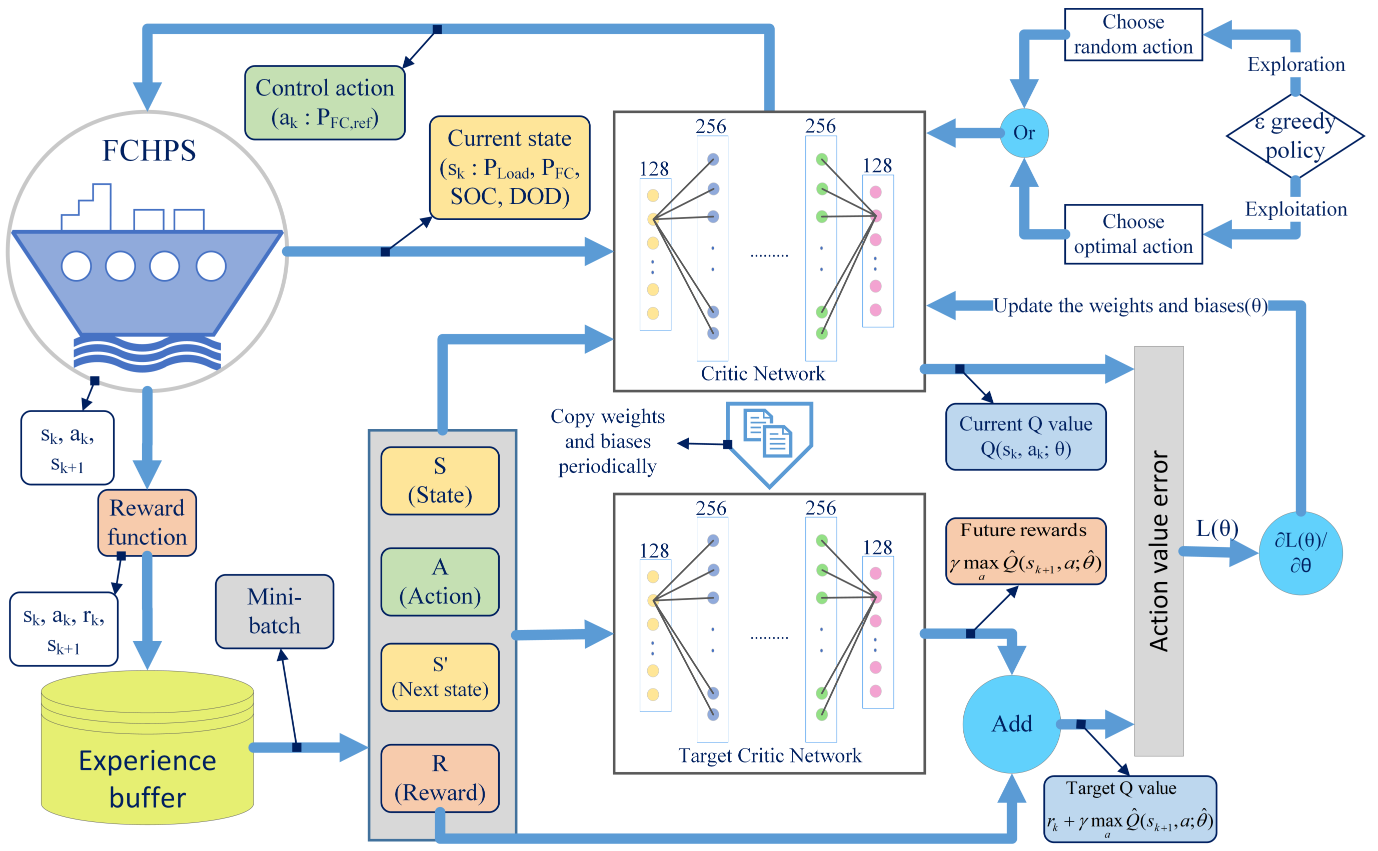


Figure 12: The framework of DDQL-based EEMS for the FCHPS.

```
Hyper-parameters: discount factor ($\gamma$), learning rate
  ($\alpha$), greedy policy ($\varepsilon$&$\Delta\varepsilon$), mini batch size ($n$)
Initialize experience buffer $D$ to capacity $N$
Initialize critic network $Q$ with random parameters $\theta$
Initialize target critic network $\hat{Q}$ with parameters
  $\hat{\theta} = \theta$
for episode(m) = 1:M do
    for iteration(k) = 1:K do
        Choose action
          $(a_k) = \begin{cases} \text{Choose random action with probability } \varepsilon \\ \textit{Otherwise } \max_a Q(s_k, a; \theta) \end{cases}$
        Execute $a_k$, observe $s_{k+1}$, and compute $r_k$
        Store transitions ($s_k$, $a_k$, $s_{k+1}$, $r_k$) in $D$
        Sample random mini-batch of transitions
          ($s_k$, $a_k$, $s_{k+1}$, $r_k$) from $D$
        Update critic network: $\theta \leftarrow \theta - \nabla_\theta L_k(\theta)$
        Update target critic network: After every C
          iterations copy the parameters
          $\hat{\theta} = \theta \Rightarrow \hat{Q} = Q$
    end
end
```

Algorithm 1: The DDQL Algorithm.

B. Soft Actor-Critic (SAC) Algorithm

The SAC algorithm is one of the recent breakthroughs within the field of reinforcement learning, proposed by Haarnoja *et al.* [53] [34]. It gained reputation as one of the most stable off-policy methods for continuous control problems. Its three main features are:

1) SAC has an actor-critic architecture with separate actor- and critic networks.
2) The actor is called stochastic actor as it is trained to maximize expected reward and entropy simultaneously, i.e., to succeed at the task while acting as randomly as possible. Entropy is a measure of the randomness of the actions. In the current form of the algorithm, Haarnoja *et al.* [34] have automated the calculation of the trade-off between maximizing the expected reward and entropy, essentially solving the problem of exploration versus exploitation dilemma. Increasing entropy leads to more exploration, which is needed to prevent the policy from converging to a local optimum.
3) SAC has an off-policy formulation that enables the reuse of previously collected data, making it more data-efficient compared with the on-policy counterparts such as PPO and SQL, where collecting a large number of data samples at every time step is necessary.

Architectures of the actor- and critic-networks used in the SAC algorithm are shown in Figure 13 and Figure 14, respectively, where μ is the mean and σ is the standard deviation of action-space's probability distribution. The actor network takes the state of the FCHPS as input and outputs the probability distribution of action space. The critic network takes the state-action pair as input and outputs the state-action value, which represents the expected cumulative reward from taking action a in state s. The SAC algorithm learns a stochastic policy, which aims to maximize both the reward and the entropy of the policy. The objective function to be maximized is

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s,a)\sim\rho_\pi} \left[r(s,a) + \alpha \mathbb{H}(\pi(\cdot \mid s)) \right], \qquad (51)$$

where $\mathbb{H}(\pi(\cdot \mid s))$ denotes the entropy of the policy π in state s. In Eqn. 51, the entropy and the sum of expected discounted rewards, given a probability distribution for visiting each state, ρ_π, should be maximized in the long term. The corresponding pseudo code is given in algorithm 2, where λ_Q, λ_π, λ are the soft update coefficients for critic network, actor network, and temperature parameter (β), respectively. A higher soft update coefficient makes the update more slowly, while a lower value speeds up the update, and it's typically set to a value between 0 and 1. The soft update coefficient influences the convergence speed and stability of the critic networks, affects the exploration-exploitation trade-off and the rate of policy changes, and leads to deterministic or stochastic policy. In this work, all soft update coefficients are set to 0.5.

```
Hyper-parameters: discount factor ($\gamma$), learning rate
  ($\alpha$), mini batch size ($n$)
Initialize experience buffer $D$ to capacity $N$
Initialize actor network $\pi$ with random parameters $\phi$
Initialize critic networks $Q_i$ with random parameters
  $\theta_i$
Initialize target critic networks $\hat{Q}_i$ with parameters
  $\hat{\theta}_i = \theta_i$
for each episode do
    for each environment step do
        Observe state ($s_k$) & sample action ($a_k$)
          based on policy $\approx\pi_\theta(\cdot \mid s_k)$
        Execute $a_k$, observe $s_{k+1}$, and calculate $r_k$
        Store transitions ($s_k$, $a_k$, $s_{k+1}$, $r_k$) in $D$
    end
    for each gradient step do
        Sample random mini-batch of transitions
          ($s_k$, $a_k$, $s_{k+1}$, $r_k$) from $D$
        Update critic network:
          $\theta_i \leftarrow \theta_i - \lambda_Q \cdot \hat{\nabla}_{\theta_i} L(\theta_i)$
        Update actor network: $\phi \leftarrow \phi - \lambda_\pi \cdot \hat{\nabla}_\phi L(\phi)$
        Update parameter $\beta$: $\beta \leftarrow \beta - \lambda \cdot \hat{\nabla}_\beta L(\beta)$
        Update target critic network: Every C
          iterations copy the parameters $\hat{\theta}_i = \theta_i$
          $\Rightarrow \hat{Q}_i = Q_i$
    end
end
```

Algorithm 2: The SAC Algorithm.

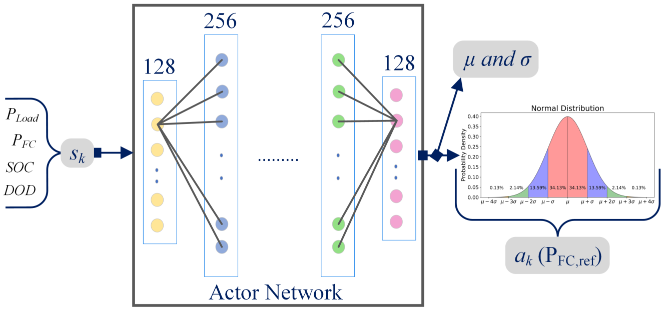


Figure 13: Actor network architecture in SAC.

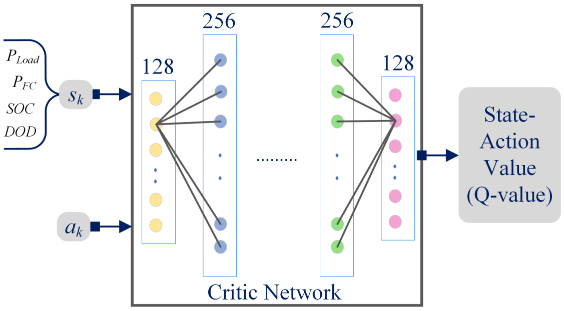


Figure 14: Critic network architecture in SAC.

C. Proximal Policy Optimization (PPO) Algorithm

The PPO algorithm proposed by [35] is an actor-critic class of RL algorithm, that uses trust region optimization, by maximizing the expected return while constraining the change in policy on each iteration. The actor network used in the PPO is similar to that of the SAC, whereas the critic network as shown in Figure 15 takes only the state of the FCHPS as input and outputs the estimated value function, which represents the expected cumulative reward starting from the current state. The objective function ($J(\phi)_t$) of the PPO aims to maximize the expected reward while maintaining stability during training by clipping the policy updates, that is,

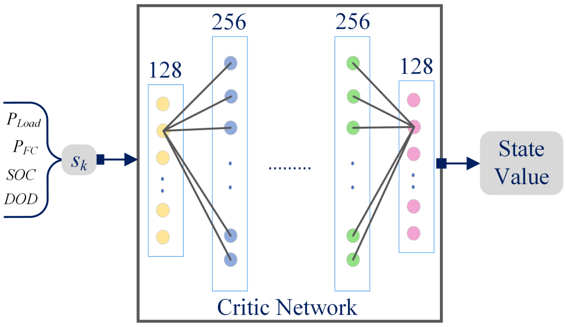


Figure 15: Critic network architecture in PPO.

$$J(\phi)_t = \min(\bar{r}(\phi)_t \hat{A}, \text{clip}(\bar{r}(\phi)_t, 1-\epsilon, 1+\epsilon)\hat{A}_t), \qquad (52)$$

where $\bar{r}(\phi)_t$ is the ratio of the output of π with the new network parameters and the old parameters, defined as

$$\bar{r}(\phi)_t = \frac{\pi_\phi(a_t|s_t)}{\pi_{\phi_{old}}(a_t|s_t)}. \qquad (53)$$

$\hat{A}$ is the advantage function, which defines how much better it is to select action a_t in state s_t over the average values of possible actions in s_t. There are several ways of approximating this function, and it is commonly applied in RL, as it reduces the variance of traditional value functions significantly. The intention of the clipped value is to constrain how much the policy is allowed to change. Thus, the probability ratio satisfies $\bar{r}(\phi)_t \in [1-\epsilon, 1+\epsilon]$, where ϵ is clipping parameter. Optimizing the advantage function gives a data-efficient on-policy RL algorithm, which is easy to implement and does not require extensive tuning of hyperparameters. The corresponding pseudo code is given in algorithm 3.

```
Hyper-parameters: discount factor ($\gamma$), clipping
  parameter ($\epsilon$), mini batch size ($n$);
Initialize actor network $\pi$ with random parameters $\phi$
Initialize critic network with random parameters $\theta$
for iteration(k) = 1:K do
    Execute the current policy $\pi_k = \pi(\phi_k)$ in
      environment for $t = 1, 2, ..., T$ time steps
    Compute the rewards ($r_t$) and advantage
      estimates ($\hat{A}_t$)
    Collect the transitions ($s_t$, $a_t$, $s_{t+1}$, $r_t$) in $D_k$
    Sample random mini-batch of transitions ($s_t$, $a_t$,
      $s_{t+1}$, $r_t$) from $D_k$
    Update actor network:
      $\phi \leftarrow \phi - \lambda_\pi \cdot \hat{\nabla}_\phi L^{CLIP}(\phi)$
      $L^{CLIP}(\phi)_t = \hat{\mathbb{E}}_t \{J(\phi)_t\}$
    Update critic network: $\theta \leftarrow \theta - \lambda \cdot \hat{\nabla}_\theta L(\theta)$
      $L(\theta)_t = \hat{\mathbb{E}}_t \left\{\min\left(V_\phi(s_t) - r_t\right)^2\right\}$
end
```

Algorithm 3: The PPO Algorithm.

V. Results

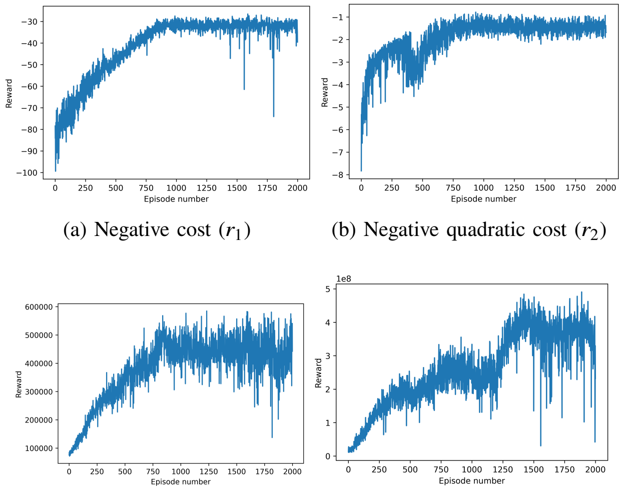
In this section, the results obtained from training and validating RL agents are presented. The RL agents are trained with hybrid models, where the linearized models are combined with nonlinear aging effects. The trained RL agents are validated with nonlinear models combined with nonlinear aging effects. The main objectives are to 1) evaluate four RFFs considered; for this purpose, three RL algorithms (DDQL, SAC, and PPO) are trained and validated on four RFFs, and 2) compare three RL algorithms and benchmark them against the rule-based energy management strategy by [49]. For the training phase of our algorithms, Intel(R) Xeon(R) Gold 6146 CPU@3.20 GHz along with 256 GB of RAM was used. Whereas, the validation of the algorithms was performed in a separate environment featuring an Intel(R) Core(TM) i9-9900K CPU@ 3.60GHz and 128 GB of RAM. The software environment included Python 3.9.7, TensorFlow 2.0, and Visual Studio Code.

A. Training

The RL agents are trained with each of the four RFFs; the hyperparameters used and the results obtained are presented. The hyperparameters for designing and training the DDQL agent are given in Table VI. Figure 16 shows the evolution of reward during training of the DDQL agent for the four RFFs. It can be observed that the reward improves approximately until episode 800 and then flattens out for NC and NQC. Though a similar trend is observed in IC, the reward is highly fluctuating. However, in IQC, the reward is highly fluctuating and it increased steeply from episode 1150 to episode 1400.

Table VI: DDQL hyperparameters.

Parameter	Description	Value
DNN parameters	Hidden layers Nodes per layer Activation function	4 [128, 256, 256, 128] ReLU
Training parameters	Optimizer Learning rate Discount factor (γ) Minibatch size	Adam 5e-4 0.99 1000
Other parameters	Evaluation interval Evaluation episodes	50 20



(a) Negative cost (r_1) (b) Negative quadratic cost (r_2)

(c) Inverse cost (r_3) (d) Inverse quadratic cost (r_4)

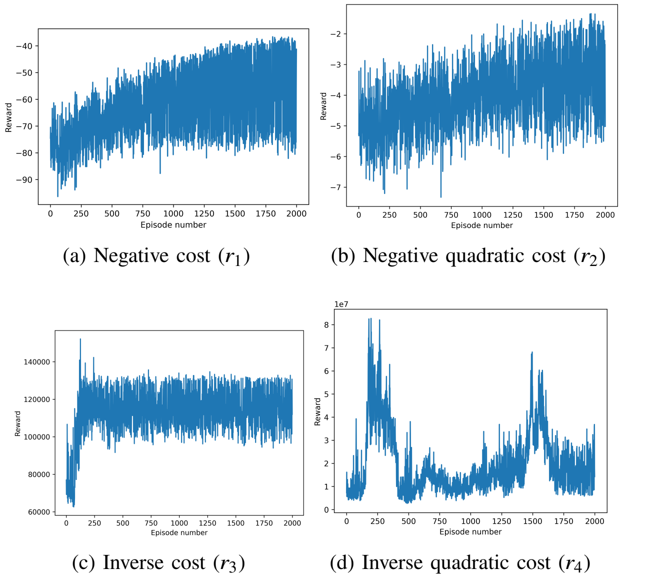
Figure 16: The evolution of rewards during training of the DDQL agent for four RFFs.

The hyperparameters for designing and training the SAC agent are given in Table VII. Figure 17 shows the evolution of reward during training of the SAC agent for the four RFFs. It can be observed that the reward is highly fluctuating, which is expected since the objective of the SAC algorithm is not only to maximize reward but also entropy.

Table VII: SAC hyperparameters.

Parameter	Description	Value
Actor network parameters	Hidden layers Nodes per layer Activation function	4 [128, 256, 256, 128] ReLU
Critic network parameters	Hidden layers Nodes per layer Activation function	4 [128, 256, 256, 128] ReLU
Training parameters	Optimizer Learning rate Discount factor (γ) Minibatch size	Adam 5e-4 0.99 1000
Other parameters	Evaluation interval Evaluation episodes	50 20

The hyperparameters for designing and training the PPO agent are given in Table VIII. Figure 18 shows the evolution of reward during training of the PPO agent for the four RFFs. The number of training episodes in the PPO is twice that of the DDQL and the SAC as the PPO is an online policy algorithm, which is data inefficient and hence requires longer training. The fluctuations in reward are much less compared to that of the DDQL and the SAC, due to the clipping of objective function to constrain the too-large policy updates. During the initial stages of training, the reward in r_1, r_2, and r_3 showed fluctuations, which eventually increased steadily and settled into a consistent average value. In contrast, the reward in r_4 decreased steadily over time and settled into a consistent



(a) Negative cost (r_1)

(b) Negative quadratic cost (r_2)

(c) Inverse cost (r_3)

(d) Inverse quadratic cost (r_4)

Figure 17: The evolution of rewards during training of the SAC agent for four RFFs.

Table VIII: PPO hyperparameters.

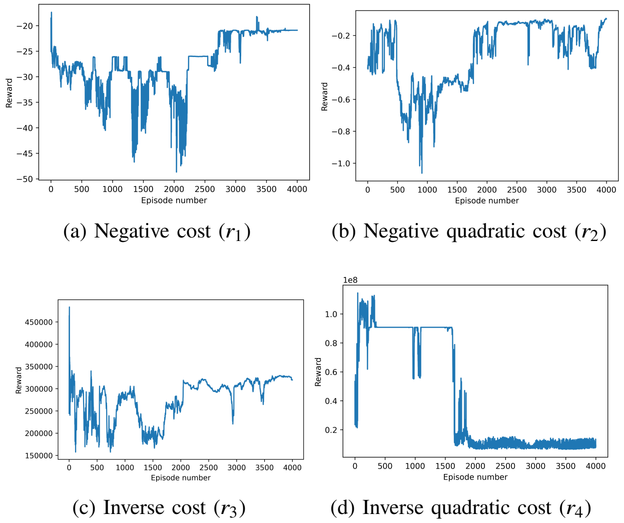
Parameter	Description	Value
Actor network parameters	Hidden layers Nodes per layer Activation function	4 [128, 256, 256, 128] ReLU
Critic network parameters	Hidden layers Nodes per layer Activation function	4 [128, 256, 256, 128] ReLU
Training parameters	Optimizer Learning rate Discount factor (γ) Minibatch size	Adam 5e-4 0.99 1000
Other parameters	Evaluation interval Evaluation episodes	50 20

average value. These observations suggest that the RFFs have a varying impact on the PPO agent's performance.

B. Validation: Comparison of RFFs

The four RFFs are validated with respective trained DDQL agents and the results are presented here. The fuel cell power, the battery power, and the battery SOC profiles are shown in Figure 19. The fuel cell power is nearly constant while the battery takes care of the load fluctuations in NC and NQC. However, both fuel cell power and battery power are highly dynamic in IC and NQC.

The operational costs including the cost of fuel, fuel cell, and battery are shown in Figure 20. The fuel cost is dominant compared to the degradation costs in the four RFFs. In NC and NQC, the degradation cost of the fuel cell is lower compared to that of the battery since the battery compensates for the fluctuating loads while fuel cell power is nearly constant. Whereas, in IC and IQC, the degradation cost of the fuel cell is higher compared to that of the battery since the fuel cell transients are expensive compared to that of the battery. The total OPEX is lowest for NC at $10.88, followed by NQC at $11.61, while IC and IQC incurred significantly higher costs at $16.24 and $23.71, respectively.



(a) Negative cost (r_1)

(b) Negative quadratic cost (r_2)

(c) Inverse cost (r_3)

(d) Inverse quadratic cost (r_4)

Figure 18: The evolution of rewards during training of the PPO agent for four RFFs.

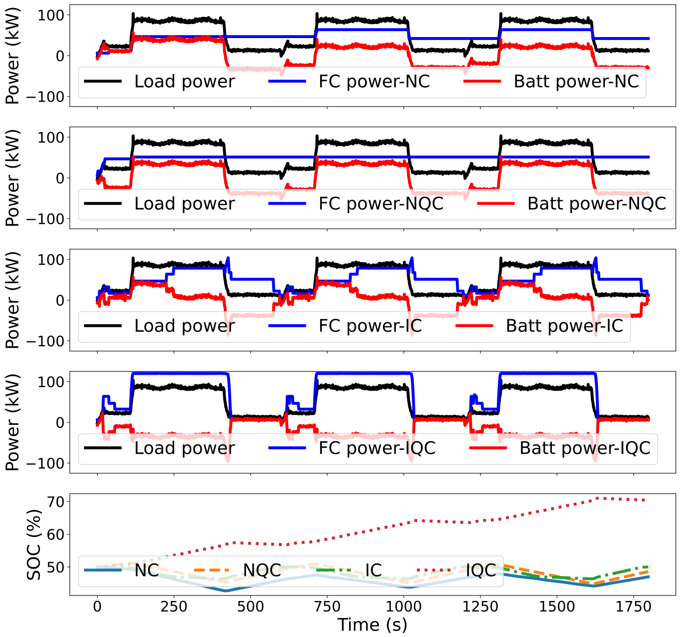


Figure 19: Validation of RFFs with DDQl agents - the fuel cell power, the battery power, and the battery SOC profiles.

The degradation parameters of the fuel cell and battery are shown in Figure 21 and Figure 22, respectively. It can be observed that NC and NQC have nearly negligible FC degradation, whereas IC and IQC have significantly higher FC degradation. Battery degradation is lowest in IC, while it is highest in IQC. However, the difference in battery degradation is not as significant as fuel cell degradation among the four RFFs.

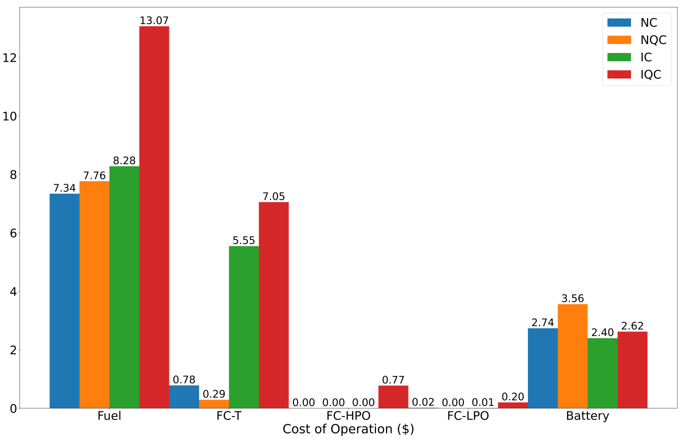


Figure 20: Validation of RFFs with DDQl agents - operational costs.

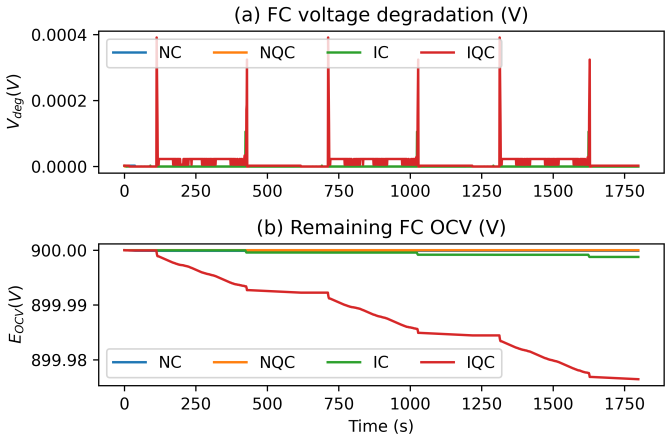


Figure 21: Validation of RFFs with DDQl agents - FC voltage degradation.

C. Validation: Comparison of RL algorithms

The RL agents for each of the three algorithms are trained with r_1(NC) and the validated results are presented for comparison and benchmarking against the rule-based EMS by [49]. The fuel cell power, the battery power, and the battery SOC profiles are shown in Figure 23. It can be observed that all the RL agents tried to minimize the fast fuel cell power transients and used the battery to compensate for the fast load changes. In the PPO, the fuel cell power is following the load power while minimizing the fuel cell transients. Whereas, in the DDQL, the fuel cell power is following the load power while remaining close to the average load power. However, in the SAC, the fuel cell power is following the load power at a much slower pace.

The operational costs, including the cost of fuel, fuel cell, and battery are shown in Figure 24. It can be observed that the fuel cost is dominant compared to the degradation costs in all the RL-based EEMS strategies. The difference among algorithms in terms of either the fuel cost or the total degradation cost is not significantly higher. The PPO agent tries to achieve a very good balance between the degradation cost of the fuel cell and the battery. Whereas, the degradation cost of the fuel cell is much lower compared to that of the battery in the DDQL and SAC agents. The total OPEX is lowest for DDQL at $10.88, followed by RB at $11.81, while SAC and PPO incurred slightly higher costs at $12.70 and $12.72, respectively.

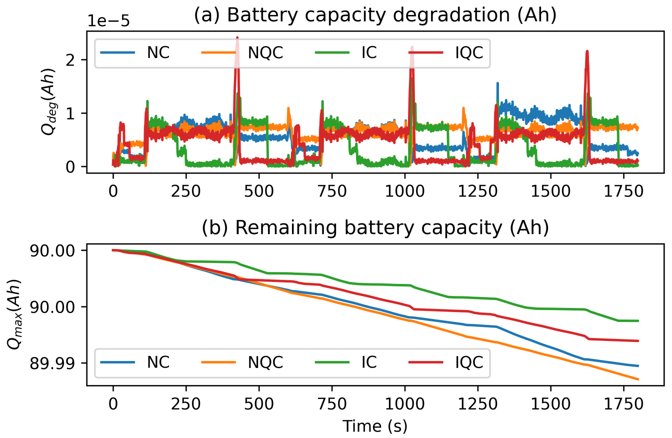


Figure 22: Validation of RFFs with DDQl agents - battery capacity degradation.

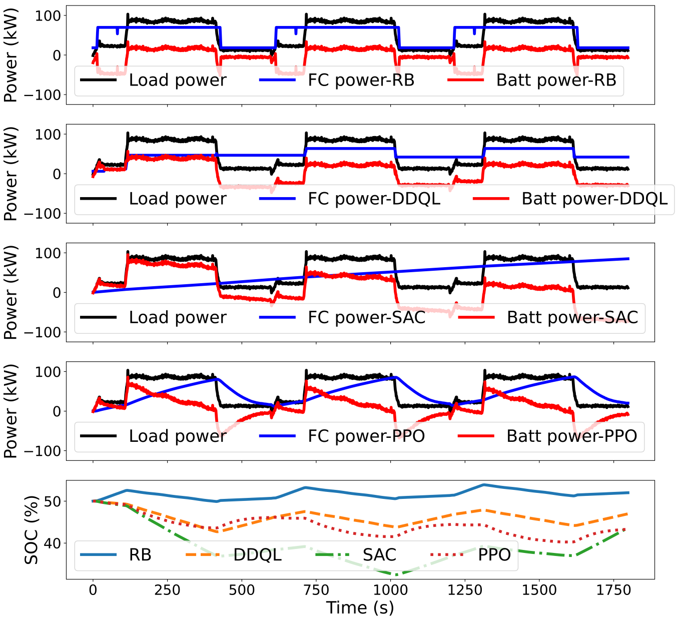


Figure 23: Comparison of RL algorithms with r_1(NC) - the fuel cell power, the battery power, and the battery SOC profiles.

The degradation parameters of the fuel cell and battery are shown in Figure 25 and Figure 26, respectively. The fuel cell degradation is lowest in rule-based followed by DDQL while it is highest in SAC followed by PPO. The battery degradation is similar both in DDQL and PPO while it is highest in SAC and lowest in rule-based.

D. Key Performance Indicators (KPIs)

In each algorithm, the RL agent is trained with each RFF and is subsequently validated with four distinct RFFs; the

This article has been accepted for publication in IEEE Journal of Emerging and Selected Topics in Industrial Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JESTIE.2023.3331230

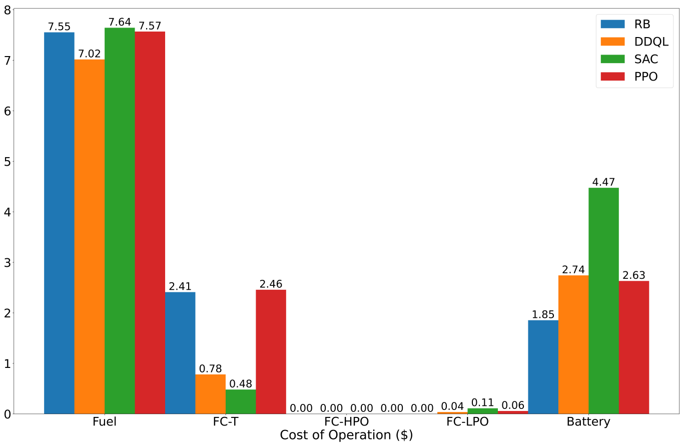


Figure 24: Comparison of RL algorithms with r_1(NC) - operational costs.

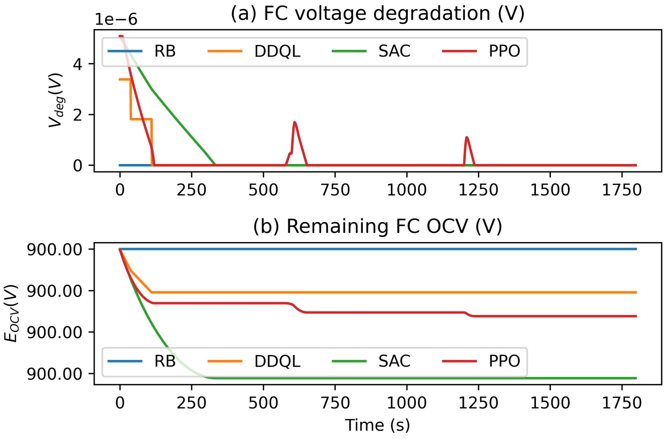


Figure 25: Comparison of RL algorithms with r_1(NC) - FC voltage degradation.

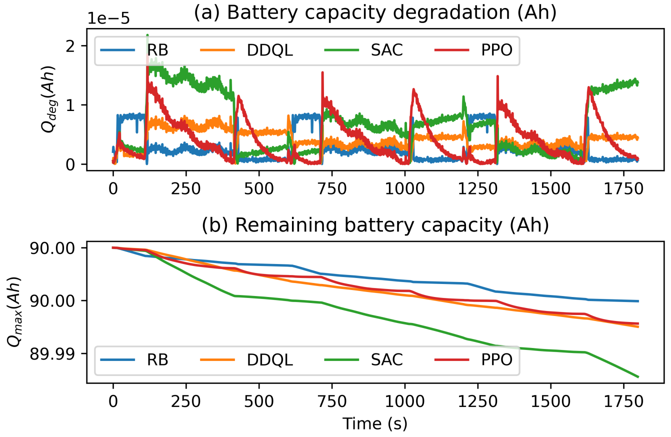


Figure 26: Comparison of RL algorithms with r_1(NC) - battery capacity degradation.

computed rewards and OPEX are defined as the KPIs. The rewards and the OPEX of the DDQL, SAC, and PPO algorithms are summarized and presented in Table IX, Table X, and Table XI, respectively. Furthermore, OPEX data pertaining to three specific cost components fuel cost (C_1), fuel cell cost (C_2), and battery cost (C_3) is presented in Table XII for the case of RFF r_1.

A careful examination of KPIs reveals several noteworthy observations. Firstly, it is evident that the choice of the RFF within each algorithm significantly impacts the OPEX across all three RL algorithms (DDQL, SAC, and PPO). RFFs r_1 and r_2 consistently result in lower OPEX values compared to r_3 and r_4. This highlights the sensitivity of OPEX to the specific RFF used for training and testing. The second observation is that the RFF r_1 consistently exhibits the lowest OPEX, followed by r_2, while r_3 and r_4 each consistently result in significantly higher OPEX across all three algorithms. This may indicate that these RFFs are more challenging from a cost perspective and may require optimization or adaptive strategies. The third critical observation is that, across all three algorithms, the KPIs reveal that RFFs r_1 and r_2 exhibit the highest reward values when they are specifically trained with RFFs r_1 and r_2 as RFFs, respectively. However, a consistent pattern is not observed for the KPIs pertaining to RFFs r_3 and r_4. The disparity in outcomes for these RFFs highlights the complexity and variability of the RL algorithm's performance when dealing with different RFFs. These observations shed light on the nuanced relationships between RL algorithm outcomes and the characteristics of RFFs, contributing to a deeper understanding of the algorithmic dynamics. Moreover, among the three algorithms DDQL, SAC, and PPO, the DDQL algorithm consistently yields the lowest overall rewards, followed by SAC, and then PPO when trained with RFFs r_1 and r_2. However, for RFFs r_3 and r_4, the differences in rewards between these algorithms are less pronounced, despite the increased OPEX values. This suggests that the algorithm's performance varies with different RFFs.

Furthermore, comparing the OPEX for RFF r_1 (NC) between rule-based and RL algorithms, it is noteworthy that the rule-based approach incurs an OPEX of \$11.81, while the DDQL algorithm achieves the lowest OPEX at \$10.88. This observation demonstrates the potential cost-saving benefits of RL algorithms compared to traditional rule-based methods. The SAC and PPO algorithms result in higher OPEX compared to DDQL for RFF r_1. SAC incurs an OPEX of \$12.70, while PPO incurs a slightly higher OPEX of \$12.72. This suggests that the choice of RL algorithm can impact OPEX outcomes for specific RFFs.

These observations underline the critical role of the RFF and the choice of RL algorithm in influencing the OPEX of the FCHPS. The findings also emphasize the potential cost-efficiency benefits of RL algorithms over rule-based methods in optimizing operational costs for specific scenarios. Furthermore, the study highlights the need for adaptive strategies in managing the variability in OPEX across different RFFs.

VI. Conclusions

In this work, state-of-the-art reinforcement learning algorithms, including double-deep Q-learning (DDQL), soft-actor critic (SAC), and proximal policy optimization (PPO)

Table IX: Comparison of RFFs with DDQL agent.

Test Train	r_1	r_2	r_3	r_4	C_1	C_2	C_3	C_{tot}
r_1	-10.9	-0.1	4.6e5	1.7e8	7.3	0.8	2.7	10.9
r_2	-11.1	-0.01	4.1e5	1.3e8	7.7	0.3	3.5	11.1
r_3	-15.6	-0.5	4.7e5	2.1e8	8.3	5.6	2.4	15.6
r_4	-23.0	-0.8	4.8e5	2.5e8	13.1	8.1	2.6	23.8

Table X: Comparison of RFFs with SAC agent.

Test Train	r_1	r_2	r_3	r_4	C_1	C_2	C_3	C_{tot}
r_1	-12.7	-0.1	3.8e5	1.2e8	7.6	0.6	4.5	12.7
r_2	-12.7	-0.1	3.8e5	1.2e8	7.7	0.6	4.5	12.7
r_3	-34.2	-1.1	1.1e5	8.4e6	14.4	12.7	7.1	34.2
r_4	-20.9	-0.3	1.9e5	2.9e7	11.2	3.4	6.2	20.9

algorithms, were introduced in the quest to develop an Energy & Emissions Management Strategy (EEMS) that does not become sub-optimal and remains optimal throughout the operational lifespan of Zero-emission Ships (ZES) powered by fuel cells and batteries. The implementation of such algorithms is not only computationally intensive but also cumbersome since it requires training with a large number of iterations and significant efforts in hyper-parameter tuning and selection of appropriate software tools. To overcome these challenges, a novel modeling approach was devised and implemented in a Python environment which is much faster than the MATLAB/Simulink environment.

In the proposed novel modeling approach, a hybrid model setup achieved by the integration of linearized polarization curve models with nonlinear aging effects is used for training reinforcement learning (RL) agents. Furthermore, the validation of these trained RL agents' performance is conducted through the utilization of nonlinear models combined with nonlinear aging effects, thereby facilitating a comprehensive assessment of our approach.

Another contribution of this work is the formulation of rewards based on realistic operational expenditure (OPEX). In order to render the degradation cost of the fuel cell more realistic, cost coefficients that vary linearly with different operational states are introduced, all the while maintaining the average value at a constant level. Additionally, the formulation of the battery degradation cost is conducted in a unique manner, informed by a comprehensive literature review, as explained in Section II-C. For the experimentation, four reward function formulations representing operational costs, r_1(NC), r_2(NQC), r_3(IC), and r_4(IQC) are explored. Each RL agent is trained with each RFF and validated with four RFFs.

Table XI: Comparison of RFFs with PPO agent.

Test Train	r_1	r_2	r_3	r_4	C_1	C_2	C_3	C_{tot}
r_1	-12.7	-0.1	3.8e5	1.3e8	7.6	2.5	2.6	12.7
r_2	-13.0	-0.1	3.6e5	1.1e8	7.6	2.3	3.1	13.0
r_3	-18.9	-0.2	2.5e5	6.4e7	9.9	4.7	4.3	18.9
r_4	-20.5	-0.3	2.1e5	4.2e7	10.7	4.8	5.1	20.5

Table XII: Comparison of RL algorithms r_1(NC) - OPEX.

Algorithm	Fuel cost	FC cost	Battery cost	Total cost
Rule-based	$7.55	$2.41	$1.85	$11.81
DDQL	$7.34	$0.82	$2.74	$10.88
SAC	$7.64	$0.59	$4.48	$12.70
PPO	$7.57	$2.51	$2.63	$12.72

The r_1(NC) has led to the lowest OPEX among the RFFs with the DDQL agent followed by r_2(NQC) which resulted in slightly higher OPEX. The r_3(IC) and r_4(IQC) resulted in significantly higher OPEX than r_1(NC). Therefore, it can be concluded that the r_1(NC) and r_2(NQC) are the most suitable RFFs to train RL agents. Among the three RL algorithms, the DDQL has led to the lowest OPEX while the SAC and the PPO have resulted in similar but slightly higher OPEX. It should be noted that the DDQL can only be implemented in discrete action space which is not suitable for real-life applications such as a zero-emission ship, where action space is continuous. Moreover, the PPO algorithm is an online policy-based algorithm, therefore, it is not data efficient and requires significantly higher data and longer training periods than its off-policy counterparts. Hence it may be concluded that the SAC algorithm with r_1(NC) is the most computationally efficient algorithm to implement as an EEMS for zero-emission ships.

The implementation of reinforcement learning (RL) algorithms in real-world hardware poses significant challenges due to complexities, uncertainties, and associated costs. To address this, a practical approach involves creating digital twins with varying fidelity levels. Low-fidelity versions simplify training and hyperparameter tuning, offering a cost-effective solution, while high-fidelity versions rigorously validate RL algorithms in real conditions while maintaining safety through predefined limits. This approach balances computational, safety, and cost considerations.

RL algorithms can be easily generalized in automotive applications, where numerous similar vehicle models are prevalent on the roads. In contrast, maritime transportation typically involves highly customized ships designed for specific use cases. In such scenarios, the development of digital twins, replicating the unique characteristics and constraints of the specific use case, and then using it to train RL algorithms could be a better strategy. This approach ensures that RL algorithms are appropriately trained for the specific use case, and optimize their performance and adaptability. Therefore, while RL algorithms offer versatile capabilities, their successful deployment in maritime transport and similar tailored applications benefits from a more customized and case-specific approach. In specific cases like this work, RL implementation aligns when the scaling and costs of fuel cells and batteries are proportional, but adjustments are essential when these vary, or if batteries serve different functions. As a future work, it would be interesting to try and implement different variants of these RL-based algorithms. For example, one idea could be to train the PPO algorithm with entropy and the SAC algorithm without entropy.

Acknowledgment

The authors would like to thank Prof. Mehdi Zadeh for his insights during the ideation of this work. This work was supported by NTNU and the Research Council of Norway through the projects: Autoferry (Cristin prj. ID: 2503813), partly through the Centre of Excellence NTNU AMOS (RCN prj. no. 223254), and SFI Autoship (RCN prj. no. 309230).

References

[1] IMO. "Fourth greenhouse gas study 2020." (2020).

[2] Sjøfartsdirektoratet. "Zero emissions in the world heritage fjords by 2026." (Sep. 1, 2023).

[3] N. P. Reddy, M. K. Zadeh, C. A. Thieme, *et al.*, "Zero-emission autonomous ferries for urban water transport: Cheaper, cleaner alternative to bridges and manned vessels," *IEEE Electrification Magazine*, vol. 7, no. 4, pp. 32–45, Dec. 2019.

[4] T. Fletcher, R. Thring, and M. Watkinson, "An energy management strategy to concurrently optimise fuel consumption & PEM fuel cell lifetime in a hybrid vehicle," *International Journal of Hydrogen Energy*, vol. 41, no. 46, pp. 21 503–21 515, Dec. 2016.

[5] A. Barré, B. Deguilhem, S. Grolleau, M. Gérard, F. Suard, and D. Riu, "A review on lithium-ion battery ageing mechanisms and estimations for automotive applications," *Journal of Power Sources*, vol. 241, pp. 680–689, Nov. 2013.

[6] M. Yue, S. Jemei, and N. Zerhouni, "Health-conscious energy management for fuel cell hybrid electric vehicles based on prognostics-enabled decision-making," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 11 483–11 491, Dec. 2019.

[7] N. Sulaiman, M. A. Hannan, A. Mohamed, E. H. Majlan, and W. R. Wan Daud, "A review on energy management system for fuel cell hybrid electric vehicle: Issues and challenges," *Renewable and Sustainable Energy Reviews*, vol. 52, pp. 802–814, Dec. 2015.

[8] N. Sulaiman, M. A. Hannan, A. Mohamed, P. J. Ker, E. H. Majlan, and W. R. Wan Daud, "Optimization of energy management system for fuel-cell hybrid electric vehicles: Issues and recommendations," *Applied Energy*, vol. 228, pp. 2061–2079, Oct. 2018.

[9] A. M. Bassam, A. B. Phillips, S. R. Turnock, and P. A. Wilson, "Development of a multi-scheme energy management strategy for a hybrid fuel cell driven passenger ship," *International Journal of Hydrogen Energy*, vol. 42, no. 1, pp. 623–635, Jan. 2017.

[10] B. Zahedi, L. E. Norum, and K. B. Ludvigsen, "Optimized efficiency of all-electric ships by dc hybrid power systems," *Journal of Power Sources*, vol. 255, pp. 341–354, Jun. 2014.

[11] R. D. Geertsma, R. R. Negenborn, K. Visser, and J. J. Hopman, "Design and control of hybrid power and propulsion systems for smart ships: A review of developments," *Applied Energy*, vol. 194, pp. 30–54, May 2017.

[12] W. Chen, K. Tai, M. W. S. Lau, *et al.*, "Optimal power and energy management control for hybrid fuel cell-fed shipboard DC microgrid," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–18, Aug. 2023.

[13] N. P. Reddy, D. Pasdeloup, M. K. Zadeh, and R. Skjetne, "An intelligent power and energy management system for fuel cell/battery hybrid electric vehicle using reinforcement learning," in *ITEC*, Jun. 2019, pp. 1–6.

[14] S. G. Wirasingha and A. Emadi, "Classification and review of control strategies for plug-in hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 1, pp. 111–122, Jan. 2011.

[15] J. Peng, H. He, and R. Xiong, "Rule based energy management strategy for a series–parallel plug-in hybrid electric bus optimized by dynamic programming," *Applied Energy*, Clean, Efficient and Affordable Energy for a Sustainable Future, vol. 185, pp. 1633–1643, Jan. 2017.

[16] Y. Li, H. He, A. Khajepour, H. Wang, and J. Peng, "Energy management for a power-split hybrid electric bus via deep reinforcement learning with terrain information," *Applied Energy*, vol. 255, pp. 113–762, Dec. 2019.

[17] H. Li, A. Ravey, A. N'Diaye, and A. Djerdir, "A novel equivalent consumption minimization strategy for hybrid electric vehicle powered by fuel cell, battery and supercapacitor," *Journal of Power Sources*, vol. 395, pp. 262–270, Aug. 2018.

[18] X. Hu, C. Zou, X. Tang, T. Liu, and L. Hu, "Cost-optimal energy management of hybrid electric vehicles using fuel cell/battery health-aware predictive control," *IEEE Transactions on Power Electronics*, vol. 35, no. 1, pp. 382–392, Jan. 2020.

[19] Amin, R. T. Bambang, A. S. Rohman, C. J. Dronkers, R. Ortega, and A. Sasongko, "Energy management of fuel cell/battery/supercapacitor hybrid power sources using model predictive control," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 1992–2002, Nov. 2014.

[20] E. Mohammadi, M. Alizadeh, M. Asgarimoghaddam, X. Wang, and M. G. Simões, "A review on application of artificial intelligence techniques in microgrids," *IEEE Journal of Emerging and Selected Topics in Industrial Electronics*, vol. 3, no. 4, pp. 878–890, Oct. 2022.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Playing atari with deep reinforcement learning," *arXiv:1312.5602 [cs]*, Dec. 2013.

[22] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *ICML*, vol. 1, Jun. 2014, pp. 605–619.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[24] H. v. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, Mar. 2016.

[25] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, “Continuous control with deep reinforcement learning,” *arXiv:1509.02971 [cs, stat]*, Jul. 2019.

[26] S. Mohan, Y. Kim, and A. G. Stefanopoulou, “Estimating the power capability of li-ion batteries using informationally partitioned estimators,” *IEEE Transactions on Control Systems Technology*, vol. 24, no. 5, pp. 1643–1654, Sep. 2016.

[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning, An Introduction*, Second Edition. MIT Press, Nov. 2018.

[28] T. Liu, Y. Zou, D. Liu, and F. Sun, “Reinforcement learning–based energy management strategy for a hybrid electric tracked vehicle,” *Energies*, vol. 8, no. 7, pp. 7243–7260, Jul. 2015.

[29] R. Xiong, J. Cao, and Q. Yu, “Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle,” *Applied Energy*, vol. 211, pp. 538–548, Feb. 2018.

[30] T. Liu, X. Hu, S. E. Li, and D. Cao, “Reinforcement learning optimized look-ahead energy management of a parallel hybrid electric vehicle,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 4, pp. 1497–1507, Aug. 2017.

[31] Y. Wu, H. Tan, J. Peng, H. Zhang, and H. He, “Deep reinforcement learning of energy management with continuous control strategy and traffic information for a series-parallel plug-in hybrid electric bus,” *Applied Energy*, vol. 247, pp. 454–466, Aug. 2019.

[32] X. Han, H. He, J. Wu, J. Peng, and Y. Li, “Energy management based on reinforcement learning with double deep q-learning for a hybrid electric tracked vehicle,” *Applied Energy*, vol. 254, pp. 113–708, Nov. 2019.

[33] A. Khalatbarisoltani, L. Boulon, and X. Hu, “Integrating model predictive control with federated reinforcement learning for decentralized energy management of fuel cell vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, Aug. 2023.

[34] T. Haarnoja, A. Zhou, K. Hartikainen, *et al.*, “Soft actor-critic algorithms and applications,” *arXiv:1812.05905 [cs, stat]*, Jan. 2019.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv:1707.06347 [cs]*, Aug. 2017.

[36] D. Fickling, “Hydrogen is a trillion dollar bet on the future,” *Bloomberg Opinion*, Dec. 2020.

[37] P. Thounthong, B. Davat, S. Rael, and P. Sethakul, “Fuel starvation,” *IEEE Industry Applications Magazine*, vol. 15, no. 4, pp. 52–59, Jul. 2009.

[38] R. Zhang, J. Tao, and H. Zhou, “Fuzzy optimal energy management for fuel cell and supercapacitor systems using neural network based driving pattern recognition,” *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 1, pp. 45–57, Jan. 2019.

[39] P. Pei, Q. Chang, and T. Tang, “A quick evaluating method for automotive fuel cell lifetime,” *International Journal of Hydrogen Energy*, TMS07: Symposium on Materials in Clean Power Systems, vol. 33, no. 14, pp. 3829–3836, Jul. 2008.

[40] H. Chen, P. Pei, and M. Song, “Lifetime prediction and the economic lifetime of proton exchange membrane fuel cells,” *Applied Energy*, vol. 142, pp. 154–163, Mar. 2015.

[41] Y. Wang, Y. Pang, H. Xu, A. Martinez, and K. S. Chen, “PEM fuel cell and electrolysis cell technologies and hydrogen infrastructure development – a review,” *Energy & Environmental Science*, vol. 15, no. 6, pp. 2288–2328, Jun. 2022.

[42] N. Harting, R. Schenkendorf, N. Wolff, and U. Krewer, “State-of-health identification of lithium-ion batteries based on nonlinear frequency response analysis: First steps with machine learning,” *Applied Sciences*, vol. 8, no. 5, pp. 8–21, May 2018.

[43] M. Koller, T. Borsche, A. Ulbig, and G. Andersson, “Defining a degradation cost function for optimal control of a battery energy storage system,” in *2013 IEEE Grenoble Conference*, Jun. 2013, pp. 1–6.

[44] B. Xu, A. Oudalov, A. Ulbig, G. Andersson, and D. S. Kirschen, “Modeling of lithium-ion battery degradation for cell life assessment,” *IEEE Transactions on Smart Grid*, vol. 9, no. 2, pp. 1131–1140, Mar. 2018.

[45] J. Wang, P. Liu, J. Hicks-Garner, *et al.*, “Cycle-life model for graphite-LiFePO4 cells,” *Journal of Power Sources*, vol. 196, no. 8, pp. 3942–3948, Apr. 2011.

[46] L. Chen, Y. Tong, and Z. Dong, “Li-ion battery performance degradation modeling for the optimal design and energy management of electrified propulsion systems,” *Energies*, vol. 13, no. 7, pp. 16–29, Jan. 2020.

[47] A. M. Bassam, A. B. Phillips, S. R. Turnock, and P. A. Wilson, “Sizing optimization of a fuel cell/battery hybrid system for a domestic ferry using a whole ship system simulator,” in *ESARS-ITEC*, Nov. 2016, pp. 1–6.

[48] S. N. M, O. Tremblay, and L. Dessaint, “A generic fuel cell model for the simulation of fuel cell vehicles,” in *2009 IEEE Vehicle Power and Propulsion Conference*, Sep. 2009.

[49] J. Han, J.-F. Charpentier, and T. Tang, “An energy management system of a fuel cell/battery hybrid boat,” *Energies*, vol. 7, no. 5, pp. 2799–2820, May 2014.

[50] O. Tremblay, L.-A. Dessaint, O. Tremblay, and L.-A. Dessaint, “Experimental validation of a battery dynamic model for EV applications,” *World Electric Vehicle Journal*, vol. 3, no. 2, pp. 289–298, Jun. 2009.

[51] N. Omar, M. A. Monem, Y. Firouz, *et al.*, “Lithium iron phosphate based battery – assessment of the aging parameters and development of cycle life model,” *Applied Energy*, vol. 113, pp. 1575–1585, Jan. 2014.

[52] L. W. Y. Chua, T. Tjahjowidodo, G. G. L. Seet, and R. Chan, “Implementation of optimization-based power management for all-electric hybrid vessels,” *IEEE Access*, vol. 6, pp. 74 339–74 354, Nov. 2018.

[53] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” *arXiv:1801.01290 [cs, stat]*, Aug. 2018.