Real-Time Balancing Control of Reconfigurable Battery Packs Using Reinforcement Learning

Ali Irshayyid, Wanqun Yang, and Jun Chen, *Senior Member, IEEE*

***Abstract*—This article proposes a reinforcement learning (RL)-based control framework for real-time state-of-charge (SOC) balancing in reconfigurable lithium-ion battery packs. Traditional battery management strategies, such as passive and active balancing, suffer from energy inefficiency and limited adaptability. Reconfigurable battery systems offer enhanced flexibility through dynamic switchable topologies. However, to fully exploit the potential of the battery pack, an intelligent control policy to determine the optimal battery configuration in real-time is needed, which is a difficult task due to the large optimization space. To address this challenge, the reconfiguration problem is formulated as a Markov Decision Process (MDP), and a policy optimization framework is proposed to guide switch decisions based on observed cell states, such as the SOC of the individual cells, in addition to the current pack configuration. The proposed RL agent operates on a discrete action space and is guided by a multiobjective reward function tailored for a battery management application. To avoid the high cost associated with physical battery training, we construct a linear surrogate model for efficient simulation-based learning. The proposed RL controller is evaluated against rule-based and greedy-based baselines, achieving superior SOC balancing performance with reduced switching frequency and improved runtime. In addition, comparison to the true optimal obtained by dynamic programming demonstrates that the proposed RL controller achieves competitive results with significantly lower computational cost.**

***Index Terms*—Reconfigurable battery packs, reinforcement learning (RL), state-of-charge (SOC) balancing, surrogate modeling.**

I. Introduction

ELECTRIC vehicles (EVs), e-bikes, and uncrewed aerial aircraft are just a few devices that frequently employ lithium-ion batteries as part of their propulsion systems [1], [2]. To achieve the high-voltage and capacity requirements, many devices use battery packs consisting of many cells connected in parallel and/or series. For example, lithium-ion battery packs in EVs are composed of hundreds of individual cells [3]. However, multicell battery packs pose significant challenges. A battery pack will always have cell imbalance, such as variance in full charge capacity because of uneven manufacturing procedures, varying degrees of aging caused by temperature fluctuations, or state-of-charge (SOC) swings [4]. This imbalance further develops over time due to different operating conditions and unavoidable irregularities in usage processes [5]. Significant consequences can arise, as the weakest cell becomes a bottleneck for the battery pack's operation, dictating its discharge capacity. Additionally, the differences among cells can significantly influence charging efficiency, capacity, and lifespan of battery packs [6], [7], [8], [9], [10], [11].

Various techniques have been developed to address this imbalance. One common method is passive balancing, which is widely adopted in the industry primarily because of its cost-effectiveness and simplicity [12]. Passive balancing works by dissipating excess energy from cells with a higher SOC as heat through shunt resistors, thereby effectively reducing their charge levels [13]. However, a major limitation of this approach is that it is a power-loss method where dissipative energy is wasted as heat, therefore reducing the overall efficiency of the battery pack. Another approach is active balancing, which employs supplementary circuits to transfer energy from cells with high charges to those with low charges [14]. While active balancing is generally more energy-efficient and less dissipative than passive methods in terms of energy transfer, it lacks energy efficiency overall because the necessary supplementary circuit constantly drains energy during operation. Furthermore, active balancing requires more complex and sophisticated circuitry, which can be costly, limiting its widespread adoption in the industry compared with passive methods. Beyond the limitations of these balancing methods, traditional battery systems with fixed series and/or parallel connections inherently struggle to handle cell differences, which directly leads to performance degradation [15]. This fixed topology lacks the capability to dynamically manage individual cell contributions, resulting in a capacity loss and reduced energy efficiency.

To overcome these limitations, the concept of a dynamically reconfigurable battery system has been proposed as a promising alternative [16], [17], [18], [19], [20], [21]. Reconfigurable batteries can dynamically change their internal topology through switching circuitry, thereby presenting a promising solution to address cell imbalance. However, they require a control policy to continuously reconfigure the series/parallel connection to achieve the desired objectives, such as maximizing the discharge efficiency or maintaining cell balancing. The complexity of this control task is amplified by the fact that battery behavior and performance are

Received 5 December 2025; revised 13 February 2026; accepted 1 April 2026. Date of publication 7 April 2026; date of current version 22 July 2026. This work was supported in part by the National Science Foundation under Award 2237317 and Award 2430374. *(Corresponding author: Jun Chen.)*

The authors are with the Department of Electrical and Computer Engineering, Oakland University, Rochester, MI 48309 USA (e-mail: aliirshayyid@oakland.edu; wanqunyang@oakland.edu; junchen@oakland.edu).

Digital Object Identifier 10.1109/TTE.2026.3681552

constantly changing based on various factors, such as the SOC, state-of-health (SoH), temperature, and current distribution. The intricate and often nonlinear interplay between these factors makes identifying the best switch configuration a challenge [19]. Moreover, as the scale of the battery pack increases, the number of potential switch combinations can become enormous, making this inherently a high-dimensional optimal control problem.

Conventional methods, such as dynamic programming [22], [23] and rule-based approaches [24], [25], face notable limitations for battery reconfiguration. On the one hand, dynamic programming or other model-based optimization approaches can be computationally expensive as the number of cells increases, limiting their practical applicability [26]. On the other hand, rule-based approaches lack generalizability and can only achieve suboptimal performance. For example, in [27], every two neighboring cells are connected in parallel when their voltage difference exceeds a predefined threshold; otherwise, they are connected in series. While easy to implement, the optimal performance is not guaranteed, therefore limiting the performance of the battery pack.

Reinforcement learning (RL), especially deep RL (DRL), has emerged as a promising alternative due to its ability to learn optimal control policies through interaction with the environment [28], [29], [30]. For example, DRL has been applied to online optimization of building energy management systems [31], traffic flow control [29], EV energy management strategies [32], charging scheduling [33], and control of fuel cell-battery systems [34]. However, training RL agents directly on physical battery hardware is impractical [35], [36]. The trial-and-error process inherent to policy learning requires repeated charging and discharging cycles, which can accelerate battery degradation. Additionally, variations in SoH across battery packs complicate generalization and may require individual tuning or retraining. More critically, deploying untrained agents on real hardware introduces safety risks, such as executing unsafe switching actions that may harm the system. On the other hand, training RL in a virtual environment also poses a significant challenge. As the number of cells increases, the action space also grows exponentially, which makes the required training episodes grow significantly as well.

Despite these challenges, recent work has begun exploring RL for reconfigurable battery control. The study in [15] proposed a DQN-based framework for reconfigurable battery pack control using OCV-based states and a Boolean connectivity array. The RL-based controller achieved 60% reduction in voltage deviation against a fixed topology battery pack. However, their approach employed a limited set of discrete switch topologies with single-objective voltage deviation minimization, without explicitly considering switching frequency or runtime optimization. Karnehm et al. [37] introduced Amortized Q-Learning for a 12-cell reconfigurable battery pack with safety-constrained action space, yet exhibited 20.3% slower SOC balancing than simple rule-based control. Another direction of research is introduced in [9], where the authors developed an RL-based event-triggered model predictive control (RLeMPC) for cell balancing in fixed-topology battery packs. The RL agent learns when to trigger computationally expensive MPC optimization rather than learning the control itself. They achieved 99% computation reduction by employing a DQN agent to determine trigger decisions (solve new MPC versus reuse stored control sequence) based on average and minimum voltages, average SOC, and pack current. However, this approach remains MPC-dependent, operates only in fixed series topologies, and does not address battery pack reconfiguration problems. These existing approaches reveal critical gaps such as limited action spaces relative to system complexity, single-objective reward formulations that ignore practical tradeoffs between balancing performance and switching costs, a lack of rigorous benchmarking against optimal control baselines, and the absence of computationally efficient surrogate models for scalable training.

To address these challenges, this article proposes an RL-based control framework for real-time reconfiguration of lithium-ion battery packs. The reconfiguration problem is formulated as a Markov decision process (MDP), and an on-policy actor–critic architecture is employed to learn switching strategies that balance the SOC across all cells while minimizing switching frequency and maintaining runtime efficiency. To reduce training time and to ensure that the battery pack provides sufficient voltage to the load, the maximum number of parallel connections is predefined. To enable safe and scalable training, we construct a linear surrogate model that emulates the electrical behavior of battery cells within a reconfigurable pack. The learned policy is deployed and validated on a physics-based simulator, and its performance is benchmarked against rule-based and greedy-based controllers. Our results demonstrate that the proposed RL framework can significantly reduce SOC imbalance by 94.6% over 135 min. Compared to dynamic programming in a short cycle, the proposed approach achieves near-optimal balancing with significantly reduced computational cost. In particular, for a short cycle of 10 min, the benchmark dynamic programming approach requires approximately four days of simulation time on our desktop computing platform. The proposed approach, hence, offers a viable solution for real-time practical deployment in advanced battery management systems. Compared to existing literature, the contributions of this article are summarized as follows.

1) A large action space consisting of 256 valid switch topologies is explored by the RL. Compared to literature, 7 topologies in [17] and 10 in [38], our approach offered significantly higher flexibility.
2) A novel computationally efficient linear surrogate model is proposed to address the computational bottleneck in existing battery reconfiguration methods [15], [17], [37], [38], which captures the SOC evolution as a discrete-time linear time-invariant system with pack current and switch configuration as inputs. The proposed surrogate model can achieve 98.68% accuracy compared to a high-fidelity electro-chemical-thermo battery simulation model, while at the same time only requiring 0.012% simulation time. It's important to note that the surrogate model does not model aging/degradation, as this article focuses on short-term control only.

3) Existing RL battery reconfiguration approaches, [15], [17], [37], lack rigorous benchmarking beyond fixed-topology or rule-based comparisons, with some [37] even underperforming a rule-based baseline. Hence, we performed comprehensive benchmark comparisons against rule-based and greedy controllers. The proposed RL-based reconfiguration controller achieves 49.7% improvement compared to the greedy controller and 74.6% improvement compared to the rule-based controller in SOC balancing.
4) To evaluate optimality, the proposed RL controller is benchmarked against dynamic programming in two settings.
   a) Dynamic programming is applied to a ten-cell pack for only two-steps due to the curse of dimensionality.
   b) To enable a longer-horizon comparison, we additionally applied dynamic programming analysis on a reduced four-cell pack, which makes a six-step (30-min) horizon tractable. In both cases, the proposed RL-based reconfiguration policy shows very comparable performance while requiring only a fraction of real-time computation.
5) Unlike existing literature, [9], [15], [37], which present only their final reward formulation without justification or tuning guidelines, sensitivity analysis is conducted to understand the influence of individual reward components, providing insight into reward function calibration to accommodate various scenarios.

The remainder of the article is organized as follows. Section II presents the dynamics of battery packs, including the cell and reconfiguration model. Section III introduces the surrogate model used for efficient RL training, while Section IV describes the proposed RL-based reconfiguration control algorithm. Section V presents simulation results, including reward sensitivity analysis and comparisons with baseline methods. The article is concluded in Section VI with future work directions.

II. Dynamics of Reconfigurable Battery Packs

A. Cell Dynamics

This study employs a cell model based on the equivalent circuit model (ECM) [39], [40], [41]. The model integrates an electrical circuit representation with a two-state thermal model to simultaneously capture the thermal and electrical dynamics of lithium-ion batteries. The electrical characteristics are represented by two *RC* pairs in series with a resistance element, as shown in Fig. 1. The cell dynamics are described by

$$\dot{V}_1 = -\frac{V_1}{R_1 C_1} + \frac{I}{C_1} \tag{1a}$$

$$\dot{V}_2 = -\frac{V_2}{R_2 C_2} + \frac{I}{C_2} \tag{1b}$$

$$v = V_{OC} - V_1 - V_2 - IR_o \tag{1c}$$

where v represents the terminal voltage, V_{OC} is the open-circuit voltage, and I is the current (positive for discharge, negative for charge). The *RC* pairs are characterized by voltages V_1, V_2, resistance R_1, R_2, and capacitance C_1, C_2, with R_o representing the series resistance. The cell SOC is governed by

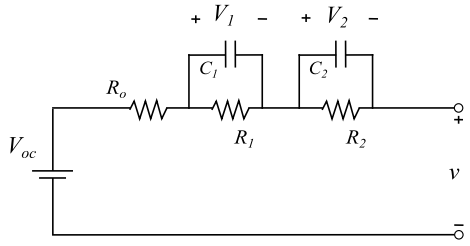


Fig. 1. ECM of a single battery cell.

$$S\dot{O}C = -\frac{\eta}{3600 C_n} I \tag{2}$$

where η is the coulombic efficiency and C_n is the nominal capacity of the cell in Amp-Hour.

The thermal behavior is captured through a two-state model incorporating core and surface temperatures

$$C_c \dot{T}_c = Q + \frac{T_s - T_c}{R_c} \tag{3a}$$

$$C_s \dot{T}_s = \frac{T_c - T_s}{R_c} + \frac{T_f - T_s}{R_u} \tag{3b}$$

where T_c and T_s represent the core and surface temperatures, respectively, T_f is the ambient temperature, C_c and C_s are the heat capacities of the core and surface, R_c represents the conduction resistance between T_c and T_s, and R_u is the convection resistance between the surface and the ambient environment. The heat generation Q is calculated as [42]

$$Q = I\,(V_{OC} - v) - I\frac{T_s + T_c}{2}\frac{dV_{OC}}{dT}. \tag{4}$$

All model parameters (V_{OC}, R_o, R_1, R_2, C_1, C_2) are functions of SOC and temperature states T_c and T_s, making the cell dynamics nonlinear. For this work, we adopt the experimentally validated parameters from [40] and [42] for a nominal cylindrical $LiFePO_4$/graphite battery cell. The electrical parameters (V_{OC}, R_o, R_1, R_2, C_1, C_2) as functions of SOC and temperature are taken from [40], while the entropic heat coefficient dV_{OC}/dT are based on the experimental measurements in [42]. It is important to note that the thermal model assumes natural air convection cooling [40], [42], which has been adopted in literature for EV study [43], [44].

Remark 1: In a reconfigurable battery pack, when switching configurations change, the current distribution among cells changes, which in turn affects individual cell temperatures through Joule heating [see (4)] [40]. Because all electrical parameters (V_{OC}, R_o, R_1, R_2, C_1, C_2) are functions of both SOC and temperature, they must be continuously updated based on the evolving thermal states (T_c, T_s). This coupling between electrical and thermal dynamics necessitates the integrated electro-thermal model to accurately predict battery pack dynamics.

B. Battery Reconfiguration

These individual cell models are integrated into a reconfigurable battery pack consisting of M cells interconnected

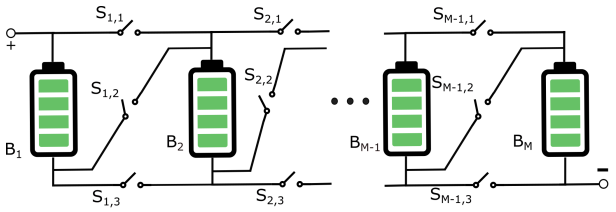


Fig. 2. Reconfigurable battery pack consisting of M cells, with a three-switch-per-connection architecture.

through a network of switches. Different switch architectures are reviewed in [45], each providing a different compromise between system flexibility and complexity. As shown in Fig. 2, a three-switch-per-connection architecture is adopted in this article, where switches enable dynamic formation of series and parallel connections between adjacent cells while also preventing open and short circuits. For example, when switch $S_{i,2}$ is connected and both switches $S_{i,1}$ and $S_{i,3}$ are disconnected, cell B_i is in a series configuration with its adjacent cells. On the other hand, when switch $S_{i,2}$ is disconnected and both switches $S_{i,1}$ and $S_{i,3}$ are connected, cell B_i forms a parallel connection with its adjacent cell B_{i+1}. Note that other switch status combinations for the three switches $(S_{i,1}, S_{i,2}, S_{i,3})$ are prohibited to prevent short or open circuits. This architecture provides sufficient degrees of freedom for topology reconfiguration while avoiding the limited flexibility of two-switch designs or the excessive complexity of four-switch arrangements.

Formally, let $\mathcal{B}$ be a battery pack with M cells $(B_1, B_2, \ldots, B_M \in \mathcal{M})$ connected through a network of switches as shown in Fig. 2, where each cell B_i has three switches $(S_{i,1}, S_{i,2}, S_{i,3})$ controlling its configuration. The switch configuration for the entire battery pack is represented by a vector $\mathbf{sw} = [sw_1, sw_2, \ldots, sw_M]$, where each element sw_i indicates the number of cells connected in parallel within the group, including cell B_i. In other words

$$sw_i = \begin{cases} 1, & \text{cell } B_i \text{ is connected in series} \\ 2, & \text{cell } B_i \text{ is part of a two-cell parallel group} \\ 3, & \text{cell } B_i \text{ is part of a three-cell parallel group} \\ \vdots & \vdots \\ k, & \text{cell } B_i \text{ is part of a } k\text{-cell parallel group.} \end{cases}$$

For example, when cell B_i is in a series configuration, $sw_i = 1$. On the other hand, when cell B_i forms a parallel connection with its adjacent cell B_{i+1}, resulting in a two-cell parallel group, we have $sw_i = sw_{i+1} = 2$. If multiple adjacent cells are connected in parallel, the corresponding sw values (for all the group) would reflect the total number of cells in that parallel group. For instance, consider a full ten-cell pack with the following configuration: cells B_1–B_3 form a three-cell parallel group, cells B_4 and B_5 are in series, cells B_6–B_7 form a two-cell parallel group, and cells B_8–B_{10} are in series, as shown in Fig. 3(a). The corresponding $\mathbf{sw}$ vector would be $[3, 3, 3, 1, 1, 2, 2, 1, 1, 1]$. As another example, Fig. 3(b) shows another pack configuration with $\mathbf{sw}$ of $[1, 1, 4, 4, 4, 4, 1, 1, 2, 2]$.

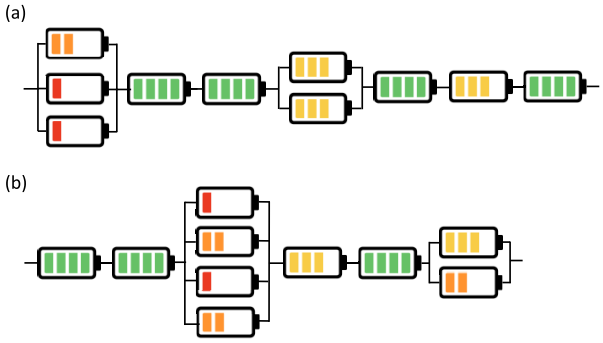


Fig. 3. Examples of battery pack reconfiguration. (a) Configuration with $\mathbf{sw} = [3, 3, 3, 1, 1, 2, 2, 1, 1, 1]$ and (b) configuration with $\mathbf{sw} = [1, 1, 4, 4, 4, 4, 1, 1, 2, 2]$.

C. Dynamics of Battery Packs

The electrical behavior of a reconfigurable battery pack is governed by Kirchhoff's current and voltage laws, which must be satisfied across all admissible topological configurations. Given a switch configuration vector $\mathbf{sw} = [sw_1, sw_2, \ldots, sw_M]$ and a total pack current demand I_{pack}, the individual cell currents $\mathbf{I} = [I_1, I_2, \ldots, I_M]^T$ can be obtained by solving a linear system that enforces the relevant circuit constraints.

For cells connected in parallel, Kirchhoff's current law requires that the total current of the parallel group equals the pack current. This yields

$$\sum_{j \in \mathcal{P}} I_j = I_{\text{pack}} \tag{5}$$

where $\mathcal{P}$ denotes the set of parallel-connected cells within a parallel group. In addition, voltage consistency is enforced across all cells within a parallel group. Specifically, for any two parallel-connected cells i and j, their terminal voltages should be equal. Therefore, the following condition is imposed:

$$\bar{V}_{\text{OC},i} - R_{o,i} I_i = \bar{V}_{\text{OC},j} - R_{o,j} I_j \tag{6}$$

or equivalently

$$R_{o,i} I_i - R_{o,j} I_j = \bar{V}_{\text{OC},i} - \bar{V}_{\text{OC},j} \tag{7}$$

where $R_{o,i}$ and $\bar{V}_{\text{OC},i} = V_{\text{OC},i} - V_{1,i} - V_{2,i}$ denote the internal resistance and "modified" open-circuit voltage of cell B_i, respectively. Please see [46] for a similar treatment on parallel-connected battery packs.

Finally, when cell B_i is connected in series, its current satisfies

$$I_i = I_{\text{pack}}. \tag{8}$$

The linear system (5), (7), and (8) can be compactly represented as

$$\mathbf{AI} = \mathbf{C} \tag{9}$$

where $\mathbf{A} \in \mathbb{R}^{M \times M}$ is the topology-dependent coefficient matrix and $\mathbf{c} \in \mathbb{R}^M$ is the constraint vector, both extracted from (5)–(8). Solving (9) yields $\mathbf{I} = \mathbf{A}^{-1}\mathbf{C}$.

Example 1: For the pack shown in Fig. 3(a) with its configuration defined as $\mathbf{sw} = [3, 3, 3, 1, 1, 2, 2, 1, 1, 1]$, the

steady-state current distribution (9) becomes (10), as shown at the bottom of the page, solving of which yields

$$I_1 = \frac{I_{pack}}{3} + \frac{R_{o,2}\left(\bar{V}_{OC,1} - \bar{V}_{OC,2}\right) + R_{o,3}\left(\bar{V}_{OC,1} - \bar{V}_{OC,3}\right)}{R_{o,1}R_{o,2} + R_{o,1}R_{o,3} + R_{o,2}R_{o,3}} \tag{11a}$$

$$I_2 = \frac{I_{pack}}{3} + \frac{R_{o,1}\left(\bar{V}_{OC,2} - \bar{V}_{OC,1}\right) + R_{o,3}\left(\bar{V}_{OC,2} - \bar{V}_{OC,3}\right)}{R_{o,1}R_{o,2} + R_{o,1}R_{o,3} + R_{o,2}R_{o,3}} \tag{11b}$$

$$I_3 = \frac{I_{pack}}{3} + \frac{R_{o,1}\left(\bar{V}_{OC,3} - \bar{V}_{OC,1}\right) + R_{o,2}\left(\bar{V}_{OC,3} - \bar{V}_{OC,2}\right)}{R_{o,1}R_{o,2} + R_{o,1}R_{o,3} + R_{o,2}R_{o,3}} \tag{11c}$$

$$I_4 = I_5 = I_8 = I_9 = I_{10} = I_{pack} \tag{11d}$$

$$I_6 = \frac{R_{o,7}I_{pack} + \bar{V}_{OC,6} - \bar{V}_{OC,7}}{R_{o,6} + R_{o,7}} \tag{11e}$$

$$I_7 = \frac{R_{o,6}I_{pack} - \bar{V}_{OC,6} + \bar{V}_{OC,7}}{R_{o,6} + R_{o,7}}. \tag{11f}$$

To sum up, the simulation of a reconfigurable pack involves, for each time step, the construction of matrices **A** and **C** by parsing the switch configuration vector **sw** to identify series and parallel substructures and applying Kirchhoff's laws accordingly. Solving the resulting system (9) yields the current distribution vector **I**, which is then used to update the battery states in accordance with (1)–(4). This formulation supports real-time computation of current distribution across topologies.

III. Surrogate Modeling of Reconfigurable Battery Pack

Implementing the cell dynamics and reconfigurable pack dynamics described in Section II allows simulation of the reconfigurable battery pack under various conditions. However, this requires solving the linear system (9) for each time step, which can significantly increase the simulation time. Such a long simulation time associated with high-fidelity physical models poses a significant challenge for RL algorithms, as it substantially increases the computational cost of policy training and delays convergence.

To overcome this issue, a surrogate model is developed to predict the evolution of cell SOC in a reconfigurable battery pack. More specifically, the following linear model is employed:

$$\mathbf{SOC}(k+1) = A \cdot \mathbf{SOC}(k) + B \cdot \mathbf{I}(k) + C \cdot \mathbf{sw}(k) \tag{12}$$

where $\mathbf{SOC}(k) \in \mathbb{R}^M$ denotes SOC for each cells at time step k, **I** is the cell current for each cell at time step k by solving (9), and **sw** is the switch configuration at time step k. By setting the time step for this linear model to 5 min, (9) needs to be solved only every 5 min. The simulation of (12) hence requires significantly less computation, suitable for RL training. Finally, matrices A, B, and C are obtained by fitting the linear model to simulation data generated from the high-fidelity model described in Section II under various initial conditions. It is important to note that while temperature is not an explicit state in (12), thermal effects are implicitly captured in A, B, and C through fitting on the high-fidelity model data.

Remark 2: In all simulations with the surrogate model, we use a time step of 5 min and assume that the current vector **I** is constant within the time step period. We use a constant-current (CC) discharge to focus only on the balancing concept, which is commonly used in the cell-balancing literature [17], [26], [38], [47]. Additionally, our framework can be used for dynamic current profiles by reducing the time step so that **I** can be updated more frequently. Validation under a dynamic current profile remains a future work direction.

More specifically, to generate training data, the high-fidelity model is simulated multiple times, each with different initial SOC, different initial temperature, and different switch configurations. The model is simulated for 5 min, and the terminated SOC is recorded. In other words, both $\mathbf{SOC}(k)$ and $\mathbf{SOC}(k+1)$ are now recorded. The matrices A, B, and C are then found through least squares. In this article, 80% of all recorded samples are randomly selected as the training set, and the remaining 20% are used as the testing set. The RMSE of the training set is 0.009051, while the RMSE of the testing set is 0.009187. Fig. 4 presents a comparison of the high-fidelity model described in Section II and the fit linear surrogate model. It can be observed that after 15 time steps, or equivalently 75 min, the linear surrogate model can still replicate the behavior of the high-fidelity model really well. Specifically, the average SOC error of all cells during the discharge is 0.0029, and the maximum error is 0.0035, demonstrating the high accuracy of this surrogate model for RL training.

IV. RL-Based Reconfiguration Control

The control of reconfigurable battery packs requires intelligent decision-making to determine the optimal switch

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ R_{o,1} & -R_{o,2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{o,2} & -R_{o,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_{o,6} & -R_{o,7} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \\ I_9 \\ I_{10} \end{bmatrix} = \begin{bmatrix} I_{\text{pack}} \\ \bar{V}_{OC,1} - \bar{V}_{OC,2} \\ \bar{V}_{OC,2} - \bar{V}_{OC,3} \\ I_{\text{pack}} \\ I_{\text{pack}} \\ I_{\text{pack}} \\ \bar{V}_{OC,6} - \bar{V}_{OC,7} \\ I_{\text{pack}} \\ I_{\text{pack}} \\ I_{\text{pack}} \end{bmatrix} \tag{10}$$

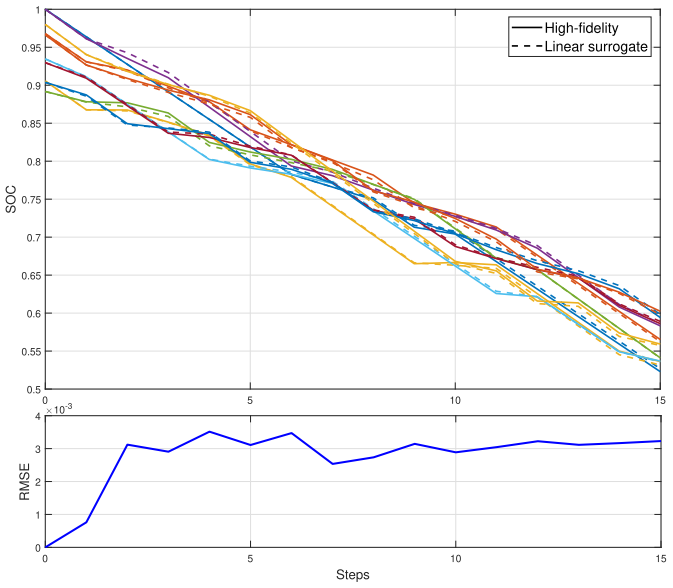


Fig. 4. Comparison of the high-fidelity model and linear surrogate model with random reconfiguration at each time step. The step size is 5 min. Top: individual cell SOC trajectories of the high-fidelity model and linear surrogate model. Bottom: the RMSE between the two models over time steps.

configuration that balances the SOC across all cells while minimizing switching frequency and maintaining efficiency. Due to the high dimensionality and nonlinearity of the system, conventional control techniques such as rule-based logic or optimization-based scheduling become impractical or computationally costly, especially as the number of cells increases. In contrast, RL offers a data-driven approach capable of learning effective policies through interactions with the environment. To leverage RL, the reconfiguration problem is formulated as an MDP, as described below.

A. RL Algorithm

The reconfigurable battery pack control algorithm is summarized in Algorithm 1. In this work, we adopt proximal policy optimization (PPO) [48] as the foundation for developing our control framework. PPO is selected over value-based methods such as deep Q-network (DQN) [49] and advantage-actor–critic (A2C) [50] due to its clipped surrogate objective, which explicitly constrains policy update magnitude. This property is particularly advantageous for our problem, where the large categorical action space (256 configurations) can lead to unstable Q value estimates in DQN or high-variance gradient updates in A2C. The empirical validation of this choice against baseline RL algorithms is provided in Section V-A. The battery reconfiguration problem is modeled as an MDP defined by a tuple $(\mathbf{S}, \mathcal{A}, P, r, \gamma)$, where the state space, $\mathbf{S}$, encodes the SOC of individual cells, action space $\mathcal{A}$ represents discrete switch configurations, r represents immediate reward, and γ is the discount factor representing a tradeoff between long-term returns and immediate gains. The policy $\pi_\theta(a|s)$ is parameterized by an actor network with parameters θ, which outputs a distribution over the action space given state s. The value function of a state s,

Algorithm 1 RL-Based Battery Pack Reconfiguration

```
Initialize policy parameters θ, value function
  parameters φ and buffer B;
for each iteration do
    while nsteps < buffer size do
        Observe state s_t;
        Sample action a_t ~ π_θ(s_t);
        Take action a_t, observe reward r_t, next state
          s_{t+1};
        Store (s_t, a_t, r_t, s_{t+1}) in buffer B;
        V_t = V_φ(s_t);
    end
    Compute advantages Â_t using (16);
    for epoch = 1, 2, ..., K do
        for each mini-batch of size N from B do
            Compute probability ratio
              r_t(θ) = π_θ(a_t|s_t) / π_{θ_old}(a_t|s_t);
            Compute actor loss function L^{CLIP+H}(θ)
              (17);
            Compute value function loss L^{VF}(13);
            Update φ using stochastic gradient descent
              to minimize L^{VF}(φ);
            Update θ using stochastic gradient ascent to
              maximize L^{CLIP+H}(θ);
        end
    end
    θ_old ← θ
end
```

$V_\phi(s)$, is estimated using a separate critic network with parameters ϕ. The agent uses two separate neural networks for actor and critic, each with three hidden layers (256–128–64, Tanh activations). The actor outputs logits over 256 actions, and the critic outputs a scalar state value. The agent collects trajectories through interaction with the proposed surrogate environment and updates both networks using mini-batch samples of the interactions. The critic is trained by minimizing the mean-squared temporal difference (TD) error

$$L^{VF}(\phi) = E\left[\left(r_t + \gamma V_\phi\left(s_{t+1}\right) - V_\phi\left(s_t\right)\right)^2\right]. \tag{13}$$

The actor is updated to maximize the expected advantage-weighted log-probability of actions as

$$\nabla_\theta J(\pi_\theta) = E_{\pi_\theta}\left[\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot A_t\right] \tag{14}$$

where $\pi_\theta(a_t|s_t)$ represents the probability of selecting action a_t in state s_t under the parameterized policy, and A_t represent the advantage function. This objective guides the policy toward selecting actions that yield higher advantage values. However, directly applying this gradient can result in instability due to excessively large updates. To enforce stability, a clipped surrogate objective that restricts the update step is applied [48]

$$L^{CLIP}(\theta) = E_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{15}$$

where $r_t(\theta) = (\pi_\theta(a_t|s_t))/(\pi_{\theta_{old}}(a_t|s_t))$ is the probability ratio between the new and old policies, and ϵ is a hyperparameter that controls the clipping range.

$\hat{A}_t$ estimates how much better an action a_t is compared to the average return in state s_t

$$A_t = r_t + \gamma V^\pi\left(s_{t+1}\right) - V^\pi\left(s_t\right). \tag{16}$$

To encourage exploration behavior of the actor network π_θ during training, an entropy bonus $\mathcal{H}(\pi_\theta(s_t)) = E\left[-\log \pi_\theta(s_t)\right]$ is added to the actor's loss $L^{CLIP}(\theta)$. The overall actor objective becomes

$$L^{CLIP+\mathcal{H}}(\theta) = L^{CLIP}(\theta) + \beta\,\mathcal{H} \tag{17}$$

where β is a weighting coefficient used to control the entropy term contribution.

B. Action Space

The action space $\mathcal{A}$ encompasses all possible switch configurations for the reconfigurable battery pack. For a pack with M cells, the action at time step t is represented as

$$a_t = \left[sw_1^t, sw_2^t, \ldots, sw_M^t\right]. \tag{18}$$

It is important to note that this work assumes that a power converter is connected between the battery pack and the load so that the output voltage is regulated at the desired operating voltage for the load [17], [51]. Additionally, to maintain a minimum nominal terminal voltage of $6 \times 3.3\,\text{V} = 19.8\,\text{V}$ (where 3.3 V is the nominal cell voltage), the number of switches allowed in parallel is limited to 4. This voltage constraint ensures adequate power delivery and prevents system undervoltage conditions. As a result, the action space is reduced from the full space of 2^{M-1} to the following constrained subset:

$$\mathcal{A} = \sum_{j=0}^{4} \binom{M-1}{j} \tag{19}$$

where $\binom{M-1}{i}$ represents the number of binary configurations with i parallel connections.

In the proposed RL framework, the actor network outputs a discrete action index $i_t \in \left\{1, 2, \ldots, \sum_{j=0}^{4}\binom{M-1}{j}\right\}$ representing a categorical choice among the valid configurations. Given the current system state, the actor outputs a softmax over all valid logits, producing a categorical distribution over all feasible action indices. The selected index i_t is then mapped to the corresponding switch configuration a_t using a predefined lookup table $\mathcal{L}$

$$a_t = \mathcal{L}(i_t). \tag{20}$$

This abstraction allows the RL agent to operate with a discrete action index while ensuring that every selected action corresponds to a physically valid and safe switch configuration, improving both learning stability and operational feasibility.

C. State Space

The state space captures the essential information required for optimal switch configuration decisions. At each time step t, the state is defined as

$$s_t = \left[SOC_1^t, SOC_2^t, \ldots, SOC_M^t, sw_1^{t-1}, sw_2^{t-1}, \ldots, sw_M^{t-1}\right] \tag{21}$$

where SOC_i^t represents the SOC of the cell B_i at time t. This state representation provides the agent with complete information about the current charge distribution across all cells in addition to the current switch connections, enabling it to make informed decisions about switch configurations to minimize SOC imbalance.

To ensure safe battery operation and prevent overdischarge, episodes terminate when any cell reaches an SOC below 0.3. This termination condition reflects realistic safety constraints in practical battery management systems while also allowing sufficient discharge depth for meaningful balancing performance evaluation.

D. Reward Function

The reward function is designed as a multiobjective function that balances SOC equalization, switching frequency, and runtime maximization. The total reward at each time step is formulated as

$$r_t = w_{SOC} \cdot R_{SOC} + w_{sw} \cdot R_{sw} + w_{time} \cdot 1 \tag{22}$$

where w_{SOC}, w_{sw}, and w_{time} are weighting factors that control the relative importance of each objective, and the constant "1" represents a fixed reward per time step for the runtime objective. The SOC balancing term encourages uniform charge distribution

$$R_{SOC} = \Delta SOC_{t-1} - \Delta SOC_t \tag{23}$$

where $\Delta SOC_t = \max_{1\le i\le M} SOC_i^t - \min_{1\le i\le M} SOC_i^t$ represents the SOC imbalance across all cells at time t. This term rewards the agent for reducing the difference between the maximum and minimum SOC values.

The switching penalty term discourages frequent switch changes and can be defined as

$$R_{sw} = -\sum_{i=1}^{M} \mathbf{1}\left[\mathcal{S}_i^{t-1} \neq \mathcal{S}_i^t\right] \tag{24}$$

where $\mathcal{S}_i^t$ represents the connection set $(S_{i,1}^t, S_{i,2}^t, S_{i,3}^t)$ between cells i and $i+1$ at time t and $\mathbf{1}[\cdot]$ is the indicator function that equals 1 when the condition is true and 0 otherwise. This term applies a negative reward for each switch state modification, to encourage a stable operation and reduce wear on switching components.

V. Results and Discussions

This section presents the performance evaluation of the proposed RL-based reconfiguration control algorithm. During training, the RL agent interacts exclusively with the data-driven surrogate model, which predicts the SOC of the pack of the next step given the switch configurations. Interacting with the surrogate model enables thousands of episodes

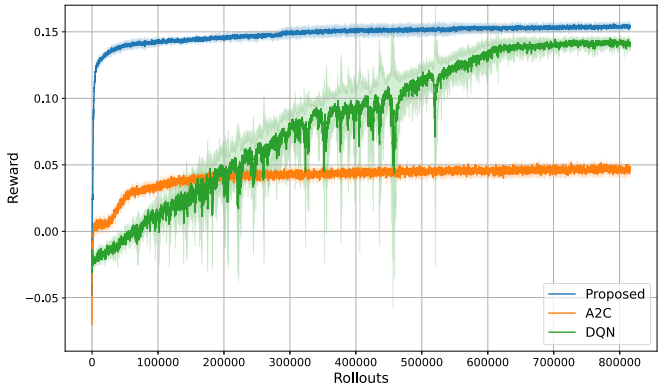


Fig. 5. Training curves of the proposed agents versus two benchmarks (A2C and DQN).

to be simulated in a fraction of the time required by the physics-based model described in Section II. Once training is complete, the trained RL agent is evaluated and tested on the physics-based model to assess its performance. To evaluate proposed RL and baseline controllers, we adopted a ten-cell reconfigurable battery pack, where cells have different amp-hour capacity due to cell-to-cell variations. Specifically, in the high-fidelity model, the nominal cell capacity of the ith cell is sampled as $C_{n,i} \in [2.0, 2.3]$ Ah, and the series resistance is modeled as $R_{o,i} = \alpha_i R_o$, where $\alpha_i \in [1.0, 1.2]$ (up to 20% variation). These heterogeneous parameters directly affect the per-cell voltage and SOC dynamics, and therefore, all reported evaluations are conducted under nonidentical cell behavior. Given the constrained configurations among cells, a ten-cell battery pack yields 256 unique configurations. Each simulation run begins with randomized initial conditions across all cells. The initial surface T_s and core temperatures T_c of each cell are randomly sampled from the range [17.5 °C, 27.5 °C], while the initial SOC values are randomly assigned between 0.8 and 1.0, representing batteries in a high charge state. The pack is being discharged under a 1A CC discharge policy. The run is terminated when one cell's SOC reaches below 0.3.

To ensure the best performance across all RL algorithms, we conducted a grid search over key hyperparameters. For the proposed approach, the search ranges were: learning rate $\{1 \times 10^{-4}, 3 \times 10^{-4}, 7 \times 10^{-4}, 1 \times 10^{-3}\}$, β $\{0.01, 0.017, 0.03, 0.06\}$, γ $\{0.95, 0.99, 1.0\}$, batch size $\{5000, 10\,000, 20\,000\}$ and number of epochs per rollout $\{4, 8, 10, 12\}$. The final hyperparameters used are learning rate 3×10^{-4}, $\beta = 0.03$, $\gamma = 1.0$, ten epochs per rollout, 10 000 steps of batch size, and 200 million total time steps. Similarly, grid-based hyperparameter search was employed for baseline algorithm's hyperparameters tuning.

A. Comparison With Baseline RL Algorithms

To evaluate the learning efficiency and policy stability of the proposed RL framework, we compare its training performance against two widely used baseline algorithms: A2C [50] and DQN [49]. All algorithms were trained using identical environmental conditions and reward structures, with a total of 800 000 rollout steps.

Fig. 5 illustrates the training curves for each method, where the solid lines represent the mean episodic reward across eight independent runs (using different random seeds), and the shaded regions indicate the standard deviation. As shown, the proposed approach consistently outperforms both A2C and DQN, achieving faster convergence and higher average rewards. Notably, the proposed agent stabilizes after approximately 400 000 rollouts, while the A2C and DQN methods either converge much more slowly or exhibit erratic performance due to their less stable update rules [52]. Specifically, DQN suffers from higher variance during training, likely due to unstable Q value estimates, particularly in high-dimensional discrete action spaces [53]. A2C performs more consistently but converges to a suboptimal plateau relative to the proposed algorithm.

B. Control Performance Evaluation

To evaluate the real-world applicability of the proposed RL controller, we deployed the optimal policy, trained by interacting with the surrogate model environment only, on the physics-based battery simulator. A total of 47 independent test scenarios were conducted, each with randomized initial conditions to assess the generalization capability and robustness of the controller under diverse operating conditions. Specifically, in each scenario, the initial SOC and temperatures (T_c and T_s) of each cell are uniformly sampled from [0.8, 1] and [17.5 °C, 27.5 °C], respectively.

The RL controller's performance was benchmarked against two classical baselines: a rule-based controller and a model-based greedy controller, detailed as follows.

1) The adopted rule-based baseline is inspired by [27], which, at each time step, identifies four pairs of adjacent cells with the lowest combined SOC values and connects them in parallel. This is based on the observations that cells with the lowest total SOC should draw less current relative to higher ones; thereby, connecting them in parallel would naturally help equalize the pack during discharge. Although simple, this method reflects practical heuristics commonly used in embedded battery management systems.
2) The model-based greedy controller applies a greedy single-step optimization. At each control step, the algorithm exhaustively evaluates all valid switch configurations by simulating a one-step-ahead SOC evolution for each configuration using the physics-based model. The switches that result in the lowest projected one-step-ahead ΔSOC across cells are selected and executed. This baseline is more computation-intensive and is considered a short-horizon near-optimal policy for comparison purposes.

Fig. 6 and Table I summarize the results of the 47-run evaluation, where "ΔSOC" is defined as the difference between the maximum and minimum individual SOCs within the pack, "$\overline{\Delta SOC}$" is the average ΔSOC across all time steps, "Runtime" denotes the time, in minutes, taken for at least one individual cell's SOC to fall below 0.3 during discharging, and "Switch change" represents the average number of changes

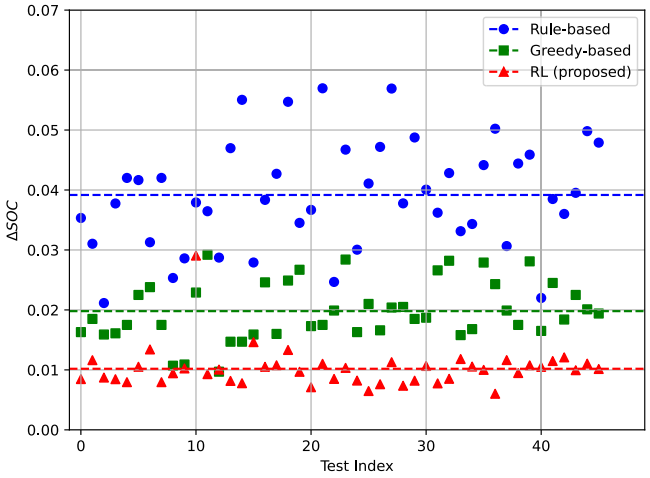


Fig. 6. Comparison of ΔSOC across 47 test runs using a rule-based, greedy controller, and the proposed RL controller. Dashed lines indicate the average ΔSOC for each method.

TABLE I
COMPARISON OF THE PROPOSED APPROACH VERSUS BASELINE CONTROLLERS

Metrics	Rule-based	Greedy-based	Proposed algorithm
ΔSOC	0.0391 (±0.008)	0.0197 (±0.004)	**0.0183** (±0.003)
$\overline{\Delta SOC}$	0.052 (±0.0113)	0.039 (±0.0144)	**0.038** (±0.0113)
Runtime	133.5 (±4.91)	105.3 (±4.09)	**135.95** (±4.495)
Switch changes	5.79 (±0.44)	3.054 (±0.57)	**2.556** (±0.78)

per time step in each connection set (S_1, S_2, S_3) among cells, where each change represents a transition between series and parallel. For example, if the current **sw** equals $[3, 3, 3, 1, 1, 2, 2, 1, 1, 1]$ and in the next step the controller decides to change **sw** to $[2, 2, 1, 1, 1, 2, 2, 1, 1, 1]$ then the number of changes for this step is 1 because $(S_{2,1}, S_{2,2}, S_{2,3})$ changed from parallel to series while all the other switches stayed the same. Taking the average of these changes during the whole test run (episode) yields the Switch change. Please note that the best value for each metric is highlighted in bold.

The proposed RL controller consistently outperforms both baselines across all four key metrics: ΔSOC, $\overline{\Delta SOC}$, runtime, and number of switch changes. For SOC balancing, the RL controller achieves the lowest ΔSOC and $\overline{\Delta SOC}$ with 0.0183 ($\sigma = 0.0033$) and 0.038 ($\sigma = 0.0113$), respectively, compared to [0.0391 ($\sigma = 0.0089$) and 0.052 ($\sigma = 0.0113$)] and [0.0197 ($\sigma = 0.0048$) and 0.039 ($\sigma = 0.0144$)] for the rule-based and greedy approaches, respectively. Notably, the RL controller maintains the lowest standard deviation, which shows robustness and stability. Additionally, it maintains a higher runtime (135.95 min) and requires the fewest average switch operations (2.556), indicating a more efficient and less aggressive switching policy. These results highlight the RL controller's ability to strike a balance between performance and operational efficiency. Figs. 7 and 8 further illustrate the SOC evolution and the ΔSOC for two test episodes using the RL controller. The SOC levels of all ten cells converge smoothly, demonstrating effective balancing without oscillatory or erratic switching behavior.

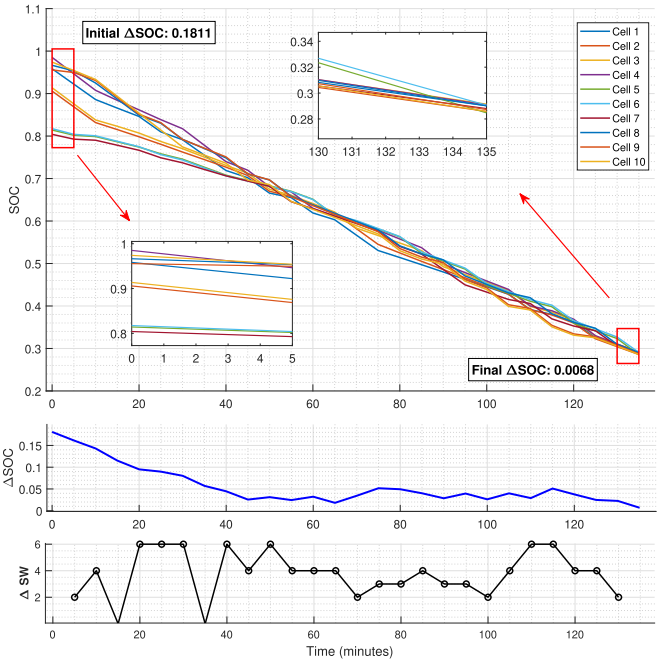


Fig. 7. Performance of RL controller balancing a ten-cell reconfigurable battery pack. Top: individual cell SOC trajectories over time. Middle: the change in ΔSOC over time. Bottom: change in switch configurations Δ**sw** at each time step.

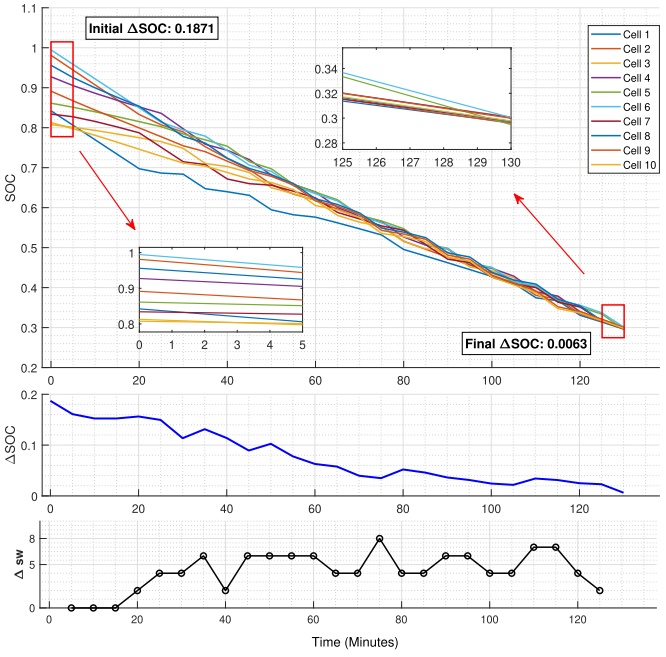


Fig. 8. Performance of RL controller balancing a ten-cell reconfigurable battery pack. Top: individual cell SOC trajectories over time. Middle: the change in ΔSOC over time. Bottom: change in switch configurations Δ**sw** at each time step.

These results collectively demonstrate the superiority of the proposed RL-based controller over both rule-based and greedy-based baselines, affirming its potential for deployment in dynamic and high-dimensional battery reconfiguration tasks.

TABLE II
PROPOSED RL CONTROLLER VERSUS TRUE MINIMUM POLICY

Model	ΔSOC (10-cells)	ΔSOC (4-cells)
RL (full episode)	0.115	0.0383
RL (matching episode)	0.104	0.0342
True Minimum	0.097	0.0294

C. Comparison With Optimal Switching Policy

To further assess the effectiveness of the proposed RL controller, we aimed to benchmark its performance against optimal switch configurations. However, exhaustively evaluating all possible switching sequences across an entire discharge cycle is computationally intractable. For a ten-cell pack with 256 possible switch combinations per time step and over 27 time steps per episode, the total number of simulations required exceeds realistic computational resources.

To overcome this, we limit the discharge cycle to only two steps, enabling a complete dynamic programming evaluation of all possible switch sequences over those two steps to find the true minimum ΔSOC. This involves an exhaustive simulation of all $256 \times 256 = 65\,536$ switch combinations per test scenario. Due to the use of a physics-based battery simulator, each such test requires approximately four days of computation time using a standard desktop computer with a 3.0 GHz processor, 32 GB of RAM, and an NVIDIA RTX 3070 Ti GPU. In total, we performed 16 such runs to get the optimal switch configurations and the true possible minimum ΔSOC. To compare against this benchmark, we evaluated two RL agents: 1) an RL agent trained on full-length episodes but evaluated in two-step episodes; and 2) an RL agent trained on two-step episodes, which matches the horizon of the optimal baseline. It is important to note that both RL agents are trained by interacting with the surrogate model only, and it takes about 7 h of training time using our standard desktop computer described earlier to train the RL agent. Additionally, to validate the proposed method for longer time steps, we investigate a four-cell battery pack where we can obtain dynamic programming results for six time steps (30 min of simulation time) within a practical amount of time. Because of the exponential increase in required simulation for dynamic programming, the simulation of seven time steps exceeds realistic computational resources; hence, six-time-step dynamic programming is implemented.

As shown in Table II, both RL agents achieve near-optimal performance in both experiments (four-cell and ten-cell battery packs). In the short-horizon ten-cell pack, the full-episode RL agent yields an average ΔSOC of 0.115, while the two-step RL agent achieves 0.104. These values are close to the minimum possible ΔSOC of 0.097 obtained through dynamic programming. Similarly, in the long-horizon four-cell pack experiment, the RL controller achieves 0.0342 and 0.0383 for the full-episode and six-step-episode RL agents, respectively, while the minimum possible average ΔSOC is 0.0294. The relatively small performance gap in both experiments highlights the RL policy's ability to approximate optimal decision-making even when trained under different scenarios. These experiments demonstrate that the proposed RL framework, despite relying on high-level approximations during training, can generalize well and perform competitively with respect to optimal control strategies in both short- and long-horizon tasks—with dramatically lower computational demands.

TABLE III
IMPACTS OF REWARD WEIGHTING

Reward Function Weights			Metrics		
w_{SOC}	w_{sw}	w_{time}	ΔSOC	Runtime (minutes)	Switch changes
1	0.01	0	0.023	135.32	1.984
1	0.03	0	0.0248	135	1.284
1	0.1	0	0.0243	135.11	**0.6259**
1	0	0	**0.0099**	132.93	4.11
1	0	0.01	0.0123	135.98	4.28
1	0	0.1	0.0107	**136.52**	4.29
1	0.03	0.1	0.0183	135.95	2.556

D. Tradeoff Analysis of Reward Function Components

To understand the influence of individual reward components, we conduct a sensitivity analysis by varying the weighting parameters of the multiobjective reward function. In this analysis, the SOC balancing term is always given primary importance ($w_{SOC} = 1$), while the switching penalty (w_{sw}) and runtime bonus (w_{time}) are adjusted to observe their effects on policy behavior. Please note that for each reward, weighting parameters, hyperparameters (learning rate, entropy coefficient, epochs, etc.) were re-tuned to ensure optimal performance under the modified reward structure. The results are summarized in Table III.

When the agent is trained with a pure SOC-focused reward ($w_{sw} = 0$, $w_{time} = 0$), it achieves the lowest ΔSOC value of 0.0099, indicating excellent balancing performance. However, this comes at the cost of aggressive switching, with an average of 4.11 switch changes per time step. This result highlights the tendency of the agent to overoptimize SOC uniformity at the expense of switching efficiency. Introducing a nonzero switching penalty (e.g., $w_{sw} = 0.01$, $w_{time} = 0$) reduces the average switch changes to 1.984, albeit with a slightly worse balancing performance ($\Delta SOC = 0.023$). Further increasing the switching weight ($w_{sw} = 0.1$) yields the lowest switching activity (0.6259), but with a corresponding drop in balancing quality ($\Delta SOC = 0.0243$). A similar trend is observed when the runtime bonus is activated. Adding a small time reward ($w_{time} = 0.01$, $w_{sw} = 0$) increases the runtime to 135.98 min and results in $\Delta SOC = 0.0123$, with switch changes increasing to 4.28. A stronger runtime emphasis ($w_{time} = 0.1$) further boosts runtime to 136.52 min, while still maintaining reasonable SOC balancing ($\Delta SOC = 0.0107$). However, the switching rate remains high in these cases. An additional configuration was tested where both the switch penalty and the runtime bonus were applied simultaneously ($w_{sw} = 0.03$, $w_{time} = 0.1$, w_{SOC}). This setup was designed to balance all three control objectives concurrently. This configuration achieves the best overall performance, with a ΔSOC of 0.0183, a runtime of 135.95 min, and an average switch change of 2.556.

VI. Conclusion

This article presents an RL control framework for real-time SOC balancing in reconfigurable lithium-ion battery packs. By modeling the reconfiguration task as an MDP, the proposed method leverages a policy optimization strategy to determine optimal switch configurations that minimize SOC imbalance while maintaining efficiency and reducing switching frequency. The ECM, integrated with thermal dynamics, provides a high-fidelity simulation environment that accurately captures the complex electro-chemical-thermal behavior of lithium-ion batteries. However, its computational expense makes it unsuitable for real-time control or large-scale training. To address this limitation, a linear surrogate model is developed to emulate the ECM's behavior with significantly reduced computational cost, enabling safe and efficient training of the RL agent. Extensive experimental validation demonstrates the superiority of the proposed RL-based controller over conventional approaches such as the rule-based controller and greedy controller. The multiobjective reward function effectively balances SOC equalization, switching efficiency, and operational runtime, with sensitivity analysis revealing optimal weighting parameters for practical deployment. Comprehensive evaluation against optimal switching policies, as obtained by dynamic programming, in short-horizon scenarios further validates the effectiveness of the approach, with the RL agent achieving performance close to the true minimum while requiring dramatically lower computational resources. With that being said, the following limitations of this work should be acknowledged. The linear surrogate model does not explicitly capture aging/degradation dynamics. Future work will focus on extending the framework to larger battery packs and developing adaptive strategies for varying operating conditions, such as SoH. In particular, experimental validation on physical battery hardware will be pursued to further validate the transfer learning capabilities and real-world applicability of the proposed approach. Additionally, the framework should be validated under time-varying discharge profiles and extended to a liquid-cooled battery pack model by refitting the surrogate model coefficients.

References

[1] A. Beaudet, F. Larouche, K. Amouzegar, P. Bouchard, and K. Zaghib, "Key challenges and opportunities for recycling electric vehicle battery materials," *Sustainability*, vol. 12, no. 14, p. 5837, Jul. 2020.

[2] J. Chen, A. Behal, and C. Li, "Active cell balancing by model predictive control for real time range extension," in *Proc. IEEE Conf. Decis. Control*, Dec. 2021, pp. 271–276.

[3] (2021). *Chevrolet Bolt EV*. [Online]. Available: https://media.chevrolet.com/media/us/en/chevrolet/vehicles/bolt-ev/2021.tab1.html

[4] Z. B. Omariba, L. Zhang, and D. Sun, "Review of battery cell balancing methodologies for optimizing battery pack performance in electric vehicles," *IEEE Access*, vol. 7, pp. 129335–129352, 2019.

[5] G. Dong, F. Yang, K.-L. Tsui, and C. Zou, "Active balancing of lithium-ion batteries using graph theory and A-star search algorithm," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2587–2599, Apr. 2021.

[6] F. Jin and K. G. Shin, "Pack sizing and reconfiguration for management of large-scale batteries," in *Proc. IEEE/ACM 3rd Int. Conf. Cyber-Physical Syst.*, Apr. 2012, pp. 138–147.

[7] J. Chen, L. Zhang, and W. Gao, "Reconfigurable model predictive control for large scale distributed systems," *IEEE Syst. J.*, vol. 18, no. 2, pp. 965–976, Jun. 2024.

[8] T. Kim, W. Qiao, and L. Qu, "A series-connected self-reconfigurable multicell battery capable of safe and effective charging/discharging and balancing operations," in *Proc. 27th Annu. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Feb. 2012, pp. 2259–2264.

[9] D. Flessner and J. Chen, "Reinforcement learning-based event-triggered model predictive control for electric vehicle active battery cell balancing," *ASME Lett. Dyn. Syst. Control*, vol. 5, no. 2, pp. 1–6, Apr. 2025.

[10] F. Altaf, "Thermal and state-of-charge balancing of batteries using multilevel converters," Dept. Signals Syst., Chalmers Tekniska Hogskola, Gothenburg, Sweden, Tech. Rep. R006/2014, 2014.

[11] J. Chen, A. Behal, Z. Li, and C. Li, "Active battery cell balancing by real-time model predictive control for extending electric vehicle driving range," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 3, pp. 4003–4015, Jul. 2024.

[12] L. Komsiyska et al., "Critical review of intelligent battery systems: Challenges, implementation, and potential for electric vehicles," *Energies*, vol. 14, no. 18, p. 5989, Sep. 2021.

[13] J. Cao, N. Schofield, and A. Emadi, "Battery balancing methods: A comprehensive review," in *Proc. IEEE Vehicle Power Propuls. Conf.*, Sep. 2008, pp. 1–6.

[14] A. Kelkar, Y. Dasari, and S. S. Williamson, "A comprehensive review of power electronics enabled active battery cell balancing for smart energy management," in *Proc. IEEE Int. Conf. Power Electron., Smart Grid Renew. Energy (PESGRE)*, Jan. 2020, pp. 1–6.

[15] F. Yang, F. Gao, B. Liu, and S. Ci, "An adaptive control framework for dynamically reconfigurable battery systems based on deep reinforcement learning," *IEEE Trans. Ind. Electron.*, vol. 69, no. 12, pp. 12980–12987, Dec. 2022.

[16] S. Ci, J. Zhang, H. Sharif, and M. Alahmad, "A novel design of adaptive reconfigurable multicell battery for power-aware embedded networked sensing systems," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2007, pp. 1043–1047.

[17] Y. Weng and C. Ababei, "AI-assisted reconfiguration of battery packs for cell balancing to extend driving runtime," *J. Energy Storage*, vol. 84, Apr. 2024, Art. no. 110853.

[18] T. Morstyn, M. Momayyezan, B. Hredzak, and V. G. Agelidis, "Distributed control for state-of-charge balancing between the modules of a reconfigurable battery energy storage system," *IEEE Trans. Power Electron.*, vol. 31, no. 11, pp. 7986–7995, Nov. 2016.

[19] S. Jeon, J. Kim, J. Ahn, and H. Cha, "Optimizing discharge efficiency of reconfigurable battery with deep reinforcement learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 11, pp. 3893–3905, Nov. 2020.

[20] A. Irshayyid and J. Chen, "GNN-based surrogate model for reconfigurable battery packs," in *Proc. IEEE Conf. Control Technol. Appl.*, Aug. 2025, pp. 71–76.

[21] S. Ci, N. Lin, and D. Wu, "Reconfigurable battery techniques and systems: A survey," *IEEE Access*, vol. 4, pp. 1175–1189, 2016.

[22] L. He, E. Kim, and K. G. Shin, "Resting weak cells to improve battery pack's capacity delivery via reconfiguration," in *Proc. 7th Int. Conf. Future Energy Syst.*, Jun. 2016, pp. 1–11.

[23] N. Lin and S. Ci, "Toward dynamic programming-based management in reconfigurable battery packs," in *Proc. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Mar. 2017, pp. 2136–2140.

[24] T. Chiueh, M.-C. Huang, K.-C. Juang, S.-H. Liang, and W. Ling, "Virtualizing energy storage management using RAIBA," in *Proc. USENIX Annu. Tech. Conf.*, 2018, pp. 187–198.

[25] C. Piao, Z. Wang, J. Cao, W. Zhang, and S. Lu, "Lithium-ion battery cell-balancing algorithm for battery management system based on real-time outlier detection," *Math. Problems Eng.*, vol. 2015, no. 1, 2015, Art. no. 168529.

[26] Y. Yang, J. He, C. Chen, and J. Wei, "Balancing awareness fast charging control for lithium-ion battery pack using deep reinforcement learning," *IEEE Trans. Ind. Electron.*, vol. 71, no. 4, pp. 3718–3727, Apr. 2024.

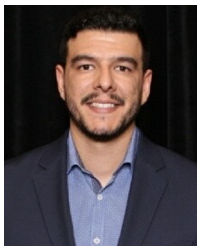
[27] S. W. Moore and P. J. Schneider, "A review of cell equalization methods for lithium ion and lithium polymer battery systems," in *Proc. SAE World Congr.*, 2001. [Online]. Available: https://doi.org/10.4271/2001-01-0959

[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[29] A. Irshayyid and J. Chen, "Highway merging control using multi–agent reinforcement learning," in *Proc. IEEE 3rd Int. Conf. Comput. Mach. Intell. (ICMI)*, Apr. 2024, pp. 1–2.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[31] E. Mocanu et al., "On-line building energy optimization using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3698–3708, Jul. 2019.
[32] Y. Li, H. He, J. Peng, and H. Wang, "Deep reinforcement learning-based energy management for a series hybrid electric vehicle enabled by history cumulative trip information," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7416–7430, Aug. 2019.
[33] Z. Wan, H. Li, H. He, and D. Prokhorov, "Model-free real-time EV charging scheduling based on deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5246–5257, Sep. 2019.
[34] 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 *Proc. IEEE Transport. Electrific. Conf. Expo (ITEC)*, Jun. 2019, pp. 1–6.
[35] J. Cao, D. Harrold, Z. Fan, T. Morstyn, D. Healey, and K. Li, "Deep reinforcement learning-based energy storage arbitrage with accurate lithium-ion battery degradation model," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4513–4521, Sep. 2020.
[36] A. Stevenson, M. Panwar, R. Hovsapian, and A. I. Sarwat, "Efficient reinforcement learning for real-time hardware-based energy system experiments," in *Proc. AAAI Symp. Ser.*, 2024, vol. 2, no. 1, pp. 153–158.
[37] D. Karnehm, W. Bliemetsrieder, S. Pohlmann, and A. Neve, "Controlling algorithm of reconfigurable battery for state of charge balancing using amortized Q-learning," *Batteries*, vol. 10, no. 4, p. 131, Apr. 2024.
[38] Y. Weng and C. Ababei, "Battery pack cell balancing using topology switching and machine learning," in *Proc. IEEE Vehicle Power Propuls. Conf. (VPPC)*, Nov. 2022, pp. 1–6.
[39] H. E. Perez, J. B. Siegel, X. Lin, A. G. Stefanopoulou, Y. Ding, and M. P. Castanier, "Parameterization and validation of an integrated electro-thermal cylindrical LFP battery model," in *Proc. Dyn. Syst. Control Conf.*, vol. 45318, New York, NY, USA, 2012, pp. 41–50.
[40] X. Lin et al., "A lumped-parameter electro-thermal model for cylindrical batteries," *J. Power Sources*, vol. 257, pp. 1–11, Jul. 2014.
[41] J. Chen, Z. Zhou, Z. Zhou, X. Wang, and B. Liaw, "Impact of battery cell imbalance on electric vehicle range," *Green Energy Intell. Transp.*, vol. 1, no. 3, Dec. 2022, Art. no. 100025.
[42] C. Forgez, D. Vinh Do, G. Friedrich, M. Morcrette, and C. Delacourt, "Thermal modeling of a cylindrical $LiFePO_4$/graphite lithium-ion battery," *J. Power Sources*, vol. 195, no. 9, pp. 2961–2968, May 2010.
[43] J. N. E. Lucero, V. A. Sujan, and S. Onori, "An experimentally validated electro-thermal EV battery pack model incorporating cycle-life aging and cell-to-cell variations," *IEEE Trans. Transport. Electrific.*, vol. 10, no. 4, pp. 8122–8136, Dec. 2024.
[44] A. Li, M. Ponchant, J. Sturm, and A. Jossen, "Reduced-order electro-thermal battery model ready for software-in-the-loop and hardware-in-the-loop BMS evaluation for an electric vehicle," *World Electric Vehicle J.*, vol. 11, no. 4, p. 75, Dec. 2020.
[45] W. Han, T. Wik, A. Kersten, G. Dong, and C. Zou, "Next-generation battery management systems: Dynamic reconfiguration," *IEEE Ind. Electron. Mag.*, vol. 14, no. 4, pp. 20–31, Dec. 2020.
[46] G. L. Plett, *Battery Management Systems: Equivalent-circuit Methods*, vol. 2. Norwood, MA, USA: Artech House, 2015.
[47] Y. Tavakol-Moghaddam, M. Boroushaki, and M. Astaneh, "Reinforcement learning for battery energy management: A new balancing approach for Li-Ion battery packs," *Results Eng.*, vol. 23, Sep. 2024, Art. no. 102532.
[48] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," Jul. 2017, *arXiv:1707.06347*.
[49] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[50] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[51] K. B. Y. Srinivasa and P. Tripura, "A DC–DC converter with battery energy storage system for electric vehicles," *J. Control Theory Appl.*, vol. 33, no. 9, pp. 61–69, 2016.
[52] A. Herrmann and H. Schaub, "A comparative analysis of reinforcement learning algorithms for Earth-observing satellite scheduling," *Frontiers Space Technol.*, vol. 4, Nov. 2023, Art. no. 1263489.
[53] H. Jia et al., "Variance reduction for deep Q-learning using stochastic recursive gradient," in *Proc. Int. Conf. Neural Inf. Process.*, 2020, pp. 634–646.



Ali Irshayyid received the B.S. degree in electrical engineering from Wasit University, Wasit, Iraq, in 2016, and the Ph.D. degree in electrical and computer engineering from Oakland University, Rochester, MI, USA, in 2024.

He is currently a Post-Doctoral Researcher at Oakland University. His research interests include reinforcement learning, optimal control, and battery management systems for transportation electrification and automotive applications.

Dr. Irshayyid served as the Chair for the IEEE Student Branch at Oakland University.

Wanqun Yang received the B.S. degree from Anhui Polytechnic University, Wuhu, China, in 2020, and the M.S. degree from Michigan State University, East Lansing, MI, USA, in 2023. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Oakland University, Rochester, MI, USA.

His research interests include model predictive control, battery thermal management systems, and advanced control algorithms for electric vehicle battery systems.

Jun Chen (Senior Member, IEEE) received the bachelor's degree in automation from Zhejiang University, Hangzhou, China, in 2009, and the Ph.D. degree in electrical engineering from Iowa State University, Ames, IA, USA, in 2014.

He was with Idaho National Laboratory, Idaho Falls, ID, USA, from 2014 to 2016, and General Motors, Milford, MI, USA, from 2017 to 2020. He joined Oakland University, Rochester, MI, USA, in 2020, where he is currently an Associate Professor with the ECE Department. His research interests include advanced control and optimization, model predictive control, artificial intelligence, and stochastic hybrid systems, with applications in intelligent vehicles, robotics, and energy systems.

Dr. Chen was a recipient of the NSF CAREER Award, the Best Paper Award from IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, the Best Paper Award from IEEE International Conference on Electro Information Technology, the Best Paper Award from IEEE Cyber Awareness and Research Symposium, the New Investigator Research Excellence Award and the Outstanding Graduate Mentor Award from Oakland University, the Publication Achievement Award from Idaho National Laboratory, the Research Excellence Award from Iowa State University, and the Outstanding Student Award from Zhejiang University.