electronics

MDPI

Article

Reinforcement Learning-Based Event-Triggered Active Battery Cell Balancing Control for Electric Vehicle Range Extension

David Flessner [1], Jun Chen [1,*] and Guojiang Xiong [2]

[1] Department of Electrical and Computer Engineering, Oakland University, Rochester, MI 48309, USA; dflessner@oakland.edu; junchen@oakland.edu
[2] Guizhou Key Laboratory of Intelligent Technology in Power System, College of Electrical Engineering, Guizhou University, Guiyang 550025, China; gjxiong1@gzu.edu.cn
* Correspondence: junchen@oakland.edu

Abstract: Optimal control techniques such as model predictive control (MPC) have been widely studied and successfully applied across a diverse field of applications. However, large computational requirements for these methods result in a significant challenge for embedded applications. While event-triggered MPC (eMPC) is one solution that could address this issue by taking advantage of the prediction horizon, one obstacle that arises with this approach is that the event-trigger policy is complex to design to fulfill both throughput and control performance requirements. To address this challenge, this paper proposes to design the event-trigger through training a deep Q network reinforcement learning agent (RLeMPC) to learn the optimal event-trigger policy. This control technique was applied to an active cell balancing controller for range extension of an electric vehicle battery. Simulation results with MPC, eMPC, and RLeMPC control policies are presented along with a discussion of the challenges of implementing RLeMPC.

Keywords: Model predictive control, event-triggered control, optimal control, reinforcement learning, active cell balancing, electric vehicle range extension

check for updates

Citation: Flessner, D.; Chen, J.; Xiong, G. Reinforcement Learning-Based Event-Triggered Active Battery Cell Balancing Control for Electric Vehicle Range Extension. *Electronics* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

Model predictive control (MPC) is an advanced control technique that has been widely studied and successfully applied in many applications, including active battery cell balancing [1,3,4,9,49–52]. MPC determines control actions by using a model of the system in combination with a cost function to determine the optimum sequence of control actions that minimizes the cost function over a future time horizon. The capability of handling constraints on the control action and state variables in the formulation of an optimal control problem (OCP) is a key feature that distinguishes MPC as a versatile tool. However, the large magnitude of computations required to solve the OCP at every time step remains a drawback for MPC implementation on embedded systems. Techniques have been investigated for reducing the computational time needed such as explicit MPC [5] and online fast MPC [6].

Another approach, event-triggered MPC (eMPC), triggers the MPC controller to determine a new set of control actions when the defined trigger conditions are met and otherwise, follow the predicted optimal control sequence [7,10–14]. The stability of the event-triggered MPC are discussed in detail in papers [47] and [46], which provide essential insights and methodologies for ensuring the stability of event-triggered MPC. In general, the stability of event-triggered MPC can be proven by Lyapunov stability theory where the cost function can be selected as the Lyapunov function. However, the design and calibration of this trigger is not trivial, so to realize the largest benefit from it, a reinforcement learning (RL) agent has been developed and trained to trigger the MPC control action. This is known as RLeMPC and has been applied in autonomous vehicle path-following problems in [15,16]. It was found in [15,16] that RLeMPC can outperform threshold-based eMPC in

key metrics related to both throughput and control performance. This work then builds on that RLeMPC formulation by using a deep neural network (DNN) to model the Q-function to determine the event-trigger for eMPC, with a particular focus on the application of active cell balancing for electric vehicle (EV) batteries. Compared to existing works such as [17] and [18] where an RL agent is used to trigger and schedule communications, this particular formulation is unique in the objective to reduce throughput rather than to save communication bandwidth.

On the other hand, the battery cell imbalance is a critical issue that limits the range of electrical vehicles (EV) [19]. In battery packs with multiple cells, the individual cells have state-of-charge (SOC) and terminal voltage variation between them that arises from manufacturing variation and uneven aging among other factors [19–21]. Discharging an individual cell below a minimum voltage results in accelerated degradation of the pack capacity and safety risks, so to prevent these conditions, any individual cell must not be discharged below a discharge voltage limit (DVL). This dynamic results in the total usable energy of the battery pack being limited by the lowest capacity cell when this cell reaches the DVL before any of the other cells while discharging. Similarly, this dynamic also plays out while charging as well where any individual cell should not be charged above a charging voltage level (CVL) to avoid cell damage and safety risks. Therefore, the charging process is then limited to whichever cell has the lowest capacity and charges to the CVL first.

Cell balancing is a technology that has arisen to address these issues [2,8,22,23,23–29,42,43]. Cell balancing methods can be classified into dissipative and nondissipative methods with the difference being whether the balancing method relies on resistive elements, ie. energy dissipative, or charge transfer, ie. nondissipative [30]. An additional dichotomy can be drawn between active and passive cell balancing that distinguishes methods that require active control from those that rely on passive circuit elements. This paper focuses on nondissipative active cell balancing with the goal of reducing the amount of capacity loss of the battery pack by redirecting charge in cells exhibiting voltage imbalance when discharging and charging to maximize the usable battery pack energy.

MPC-based active cell balancing has also been widely studied in the literature [1,22,31,35–39]. In [1], an MPC controller was developed and evaluated for active cell balancing to extend the range capability of EV batteries. Multiple cost function formulations were evaluated to determine a cost function that increases the range of the vehicle while minimizing throughput. It was concluded in [1] that all the cost function formulations tested resulted in high computation reduction, but the cost function based on minimizing tracking error of each cell voltage was the most robust against model mismatch and load disturbance. Therefore, in this paper, we focus on the development of the tracking MPC into RLeMPC with the addition of an event trigger for the OCP driven by a deep RL agent that determines when to trigger. This development is expected to improve upon the previous MPC-based approach through decreasing required throughput by reducing the amount of times the OCP must be solved while maintaining similar range extension performance. It is also expected to improve the design of the event-trigger by replacing the conventional threshold-based policy with a DNN.

The remainder of the paper is organized as follows. Section 2 describes the framework of RL and MPC, while Section 3 discusses how active cell balancing can be formulated into the RLeMPC framework. Section 4 presents simulation results from applying these techniques to cell balancing and Section 5 concludes the paper.

2. Preliminary on RL and MPC

2.1. Reinforcement Learning

The RL paradigm [32,33] operates on a state-action framework where at time step t, the state of the environment, s_t, is observed by an agent which then selects an action a_t to take. After the action is taken, a scalar reward is given to the agent depending on the reward function denoted $r(s_t, a_t)$. The goal of the agent is to learn an optimal policy

$\pi^* : s \rightarrow a$ that maximizes the expected cumulative future rewards which can be described as

$$G = E_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \tag{1}$$

where $r_t = r(s_t, a_t)$ and the scalar $\gamma \in [0, 1]$ is the discount factor that can reduce the value of future expected rewards. In order to learn the optimal policy π^*, the agent interacts with the environment to learn which actions to take at given state s_t in order to maximize the expected cumulative reward G.

To measure the value of the agent being in state s, a state value function $V_\pi(s)$ can be defined as the value of the state s under the policy π, which is the expected return starting from s following policy π. This can be expressed as

$$V^\pi(s) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^t r_{t+k} \,\middle|\, s_t = s \right] \tag{2}$$

Alternatively, the state-action value function $Q_\pi(s, a)$ of a state-action pair (s, a) can be defined as the expected return starting from s followed by action a and then policy π. This can be expressed as

$$Q^\pi(s, a) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\middle|\, s_t = s, a_t = a \right] \tag{3}$$

Through agent interaction with the environment, the Q-function can be learned, and once it is approximately known, the optimal Q-function $Q^*(s, a) = \max_\pi Q^\pi(s, a)$ is the Q-function with a policy applied to maximize the Q value. This policy π^* can be defined as

$$\pi * (s) = \arg\max_a Q * (s, a) \tag{4}$$

The RL agent will then be able to execute the optimal policy π^* by selecting the action that results in the highest Q value for a given state.

To restate, the optimal policy π^* can be found by learning the optimal Q-function $Q^*(s, a)$. The objective of RL training is then to learn $Q^*(s, a)$, which is also referred as as Q-learning. With this method, the RL agent can learn the optimal Q-function exclusively by interacting with the environment. Notably, no model of the system dynamics is required for the agent to learn the optimal policy. However, in exchange for this benefit, ample training time is required for the RL agent to learn Q^*.

2.2. Deep Neural Network Based RL

In order to decide an action, the RL agent requires a method to generalize the state-action value function over any state and action. For this implementation, a deep neural network (DNN) was used with the environment state variables as inputs to the network, and the expected, discounted cumulative future reward of each action, the Q-value, at the given state as the output. This implementation of RL is known as Deep Q-Learning because the DNN models the state-action value function or $Q(s, a)$ [34,40]. Consequently, the DNN can be more specifically described as a Deep Q Network (DQN).

To train the DQN, double Q-learning and batch update methods were used. For each update, a batch of experiences (s_i, a_i, r_i, s_i') of size M is pulled from memory to use. For double Q-learning, two sets of network weights and biases are used, denoted as ϕ for the critic network parameters and ϕ_t for the target network parameters. Depending on the state, ϕ is used to determine the next action with probability $1 - \epsilon$. Otherwise, a random

action is selected. After the action is taken, ϕ is updated according to a target values y_i. This target is determined with the double Q-learning and batch methods by

$$a_{max} = \arg\max_{a'} Q(s_i', a', \phi) \tag{5a}$$

$$y_i = r_i + \gamma Q_t(s_i', a_{max}; \phi_t) \tag{5b}$$

For double Q-learning, ϕ is used to select the action a_{max} that ϕ_t will use to determine the target. These target values y_i are then included in a loss function that compares the target value that was observed to the current critic network estimate. Stochastic gradient descent can be applied to this loss function to determine a new ϕ that minimizes L depending on the gradient of the loss function and the learning rate.

$$L = \frac{1}{2M} \sum_{i=1}^{M} (y_i - Q(s_i, a_i; \phi))^2 \tag{6}$$

The new weights and biases ϕ are then set in the behavior network for the agent to select the next action. After a set amount of training time steps, the target network parameters ϕ_t are set to the behavior network parameters ϕ in order to increase the stability of learning and reduce overestimates of Q.

The goal of training the network is to adjust the DQN parameters to closely approximate the actual Q^* function to select the action with the highest expected reward. Methods such as epsilon greedy can be used to force the agent to explore before it has had much experience with the environment and gradually transition the agent to exploit the environment once enough experience has be gained to consistently achieve high rewards.

2.3. Event-Triggered MPC

Generally, event-triggered MPC (eMPC) builds off a conventional time-triggered MPC controller by adding an event-trigger layer and changing how the control command is computed in the absence of an event. Mathematically, for a system described by the following dynamics

$$\zeta_{t+1} = f(\zeta_t, u_t) \tag{7}$$

where $\zeta_t \in \mathbb{R}^n$ is the system state at discrete time t and u_t is the controller output. The MPC controller calculates the optimal control sequence U_t and optimal state sequence ζ_t to minimize a cost function J_{mpc} over a defined prediction horizon p through solving an OCP defined as

$$\min_{Z_t, U_t} \quad \sum_{k=1}^{p} J_{mpc}^k(Z_t, U_t) \tag{8a}$$

$$\text{s.t.} \quad \zeta_t = \hat{\zeta}_t \tag{8b}$$

$$\zeta_{t+k} = f(\zeta_{t+k-1}, u_{t+k-1}), \quad 1 \le k \le p \tag{8c}$$

$$\zeta_{min} \le \zeta_{t+k} \le \zeta_{max}, \quad 1 \le k \le p \tag{8d}$$

$$u_{min} \le u_{t+k} \le u_{max}, \quad 1 \le k \le p-1 \tag{8e}$$

$$\Delta_{min} \le u_{t+k} - u_{t+k-1} \le \Delta_{max}, \quad 1 \le k \le p-1 \tag{8f}$$

where U_t and Z_t are defined as $U_t = \{u_t, u_{t+1}, ..., u_{t+p-1}\}$ and $Z_t = \{\zeta_{t+1}, \zeta_{t+2}, ..., \zeta_{t+p}\}$. The optimization is subject to the current estimate of the state $\hat{\zeta}_t$ (8b), subsequent states that only depend on the previous state and control action taken (8c), and constraints that are applied to the state and control action (8d-f). Conventionally, this OCP is solved at every time step where the first control action u_t is applied then the rest of the control sequence U_t is not used. eMPC in contrast only solves the OCP when the event conditions are met, denoted by γ_{ctrl} where $\gamma_{ctrl} = 1$ when the event conditions are met and otherwise $\gamma_{ctrl} = 0$.

When $\gamma_{ctrl} = 0$, the controller applies the predicted optimal control sequence U_{t_1} computed at the previous trigger at time t_1. This can be described as

$$u = \begin{cases} \text{Solution of (8)} & \text{if } \gamma_{ctrl} = 1 \\ U_{t_1}(k+1) & \text{Otherwise.} \end{cases} \tag{9}$$

To summarize, the primary new features that distinguish eMPC from conventional MPC are that the trigger γ_{ctrl} must be changed from a consistent periodic time trigger to event based and that it applies the predicted optimal control sequence U_t calculated until the next event is triggered. The trigger policy then becomes the focus of the design which can be described as

$$\gamma_{ctrl} = \pi(Z_{t_1}, \hat{\zeta}_t | \theta) \tag{10}$$

where Z_{t_1} is the optimal state sequence computed at the last event at time t_1, $\hat{\zeta}_t$ is the current state feedback, and θ is calibration parameters for the trigger. Typically, the event is based on the error between the optimal predicted state calculated at the last event and the current state feedback or estimation such that if the error is large between the state prediction ζ_t from the prediction sequence Z_{t_1} and current actual state $\hat{\zeta}_t$, then trigger the MCP to use the more accurate feedback to determine a new control sequence. This type of event-trigger condition can be described as

$$\left\| \zeta_t - \hat{\zeta}_t \right\| \geq \bar{e} \tag{11}$$

where $\bar{e} \in \theta$ is a calibrate-able error threshold. Details for eMPC are described in Algorithm 1 and Fig. 1.

Algorithm 1 Event-Triggered MPC [41]

procedure EMPC($\hat{\zeta}$, k, U_{t_1}, Z_{t_1}, p)
  $k \leftarrow k + 1$;
  $\gamma_{ctrl} \leftarrow$ computing (11);
  if $\gamma_{ctrl} = 1$ **then**
    $k \leftarrow 1$;
    $(Z_t, U_t) \leftarrow$ Solving OCP (16);
    $u \leftarrow U_t(1)$;
    $\zeta \leftarrow Z_{t_1}(1)$;
    $U_{t_1} \leftarrow U_t$;
    $Z_{t_1} \leftarrow Z_t$;
  else
    $u \leftarrow U_{t_1}(1)$;
    if $k <= p$ **then**
      $\zeta \leftarrow Z_{t_1}(k)$
    else
      $\zeta \leftarrow Z_{t_1}(p)$
    end if
  end if
  return u, ζ, U_{t_1}, Z_{t_1}, k
end procedure

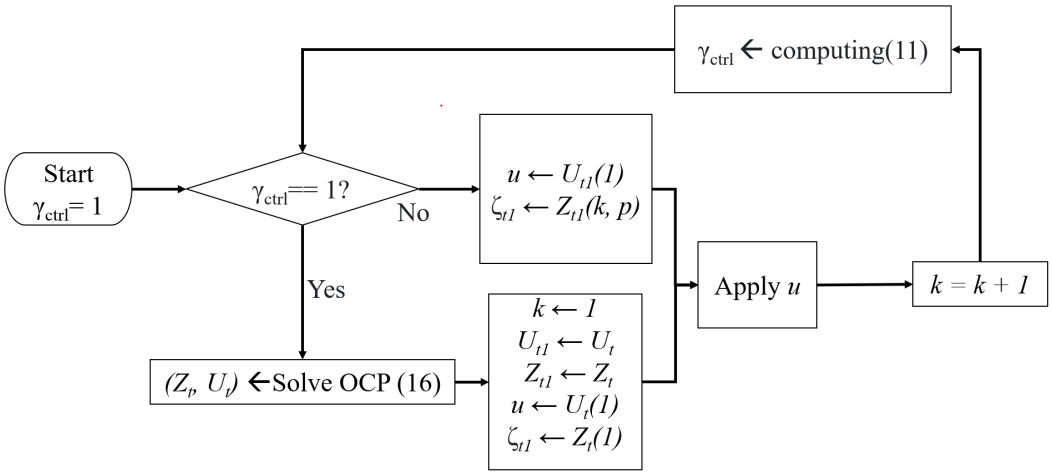

Figure 1. Flowchart for the Event-Triggered MPC (eMPC) in Algorithm 1.

However, the challenge associated with (11) is uncovered then because the analytical form of the MPC closed loop system, especially for nonlinear functions and models is difficult to determine. This results in event-trigger policy designs which are usually problem specific and non-trivial. To solve this challenge, an RL agent with a DQN is proposed to learn the optimal event-trigger policy π^* without a model of the closed-loop system dynamics.

3. Problem Formulation

3.1. Active Battery Cell Balancing Control

A common equivalent circuit model of a lithium ion cell can be used to model the SOC, relaxation voltage, and terminal voltage of each cell [44,45]. The cell model can be described as

$$\dot{s}^n = -\eta^n \frac{i^n}{C^n} \tag{12a}$$

$$\dot{V}_p^n = -\frac{V_p^n}{R_p^n C_p^n} + \frac{i^n}{C_p^n} \tag{12b}$$

$$y^n = V_{oc}^n - V_p^n - i^n R_o^n \tag{12c}$$

where the superscript n is the nth cell, s^n is the cell SOC, η^n is the cell coulombic efficiency, C^n is the cell capacity in amp hours, V_p^n is the relaxation voltage over R_p^n, V_{oc}^n is the open circuit voltage, y^n is the terminal voltage, and i^n is the cell current. Positive i^n indicates discharging the battery while negative i^n indicates charging. The variables V_{oc}^n, R_o^n, R_p^n, and C_p^n are all dependent of s^n resulting in a nonlinear cell model.

This model can then be discretized using Euler's method for a model that can be implemented digitally as

$$s_{k+1}^n = s_k^n - \eta^n \frac{T_s}{C^n} i_k^n \tag{13a}$$

$$V_{p,k+1}^n = V_{p,k}^n - \frac{T_s}{R_p^n C_p^n} V_{p,k}^n + \frac{T_s}{C_p^n} i_k^n \tag{13b}$$

$$y_k^n = V_{oc,k}^n - V_{p,k}^n - i_k^n R_o^n \tag{13c}$$

where T_s was set to 1 s. To simulate imbalance, a probability distribution can be used to apply random variation to the C^n, R_o^n, and C_p^n terms. Table 1 lists example variations of these parameters that would later be used for simulation in this paper.

Table 1. Cell Imbalance Parameters at 100% SOC. Ah: Amp-hour; mΩ: milliohms; kF: kilofarads.

Parameter	Unit	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
C	Ah	62.87	60.00	66.61	56.73	61.66
R_p	mΩ	6.40	5.66	5.47	6.68	6.36
C_p	kF	153.7	177.8	175.9	168.9	150.4
R_o	mΩ	1.49	1.27	1.41	1.51	1.53

Finally, the SOC and relaxation voltage states can be grouped together as $\zeta^n := [s^n, V_p]^T$ where $.^T$ denotes a matrix or vector transpose. The battery pack state can be written as

$$\zeta^n_{k+1} = f^n(\zeta^n_k, i_k + u^n_k) \tag{14a}$$

$$y^n_k = g^n(\zeta^n_k, i_k + u^n_k) \tag{14b}$$

where u^n_k is the balancing current applied to the cell as shown in Fig. 2. Define $\zeta = [\zeta^1, \zeta^2, ..., \zeta^N]^T$ as the state vector for the battery pack and y to be the terminal voltage of the battery pack, then

$$\zeta_{k+1} = \begin{bmatrix} f^n(\zeta^1_k, i_k + u^1_k) \\ f^n(\zeta^2_k, i_k + u^2_k) \\ \vdots \\ f^n(\zeta^N_k, i_k + u^N_k) \end{bmatrix} \tag{15a}$$

$$y_k = \sum_{m=1}^{N} y^n_k = \sum_{m=1}^{N} g^n(\zeta^n_k, i_k + u^n_k) \tag{15b}$$

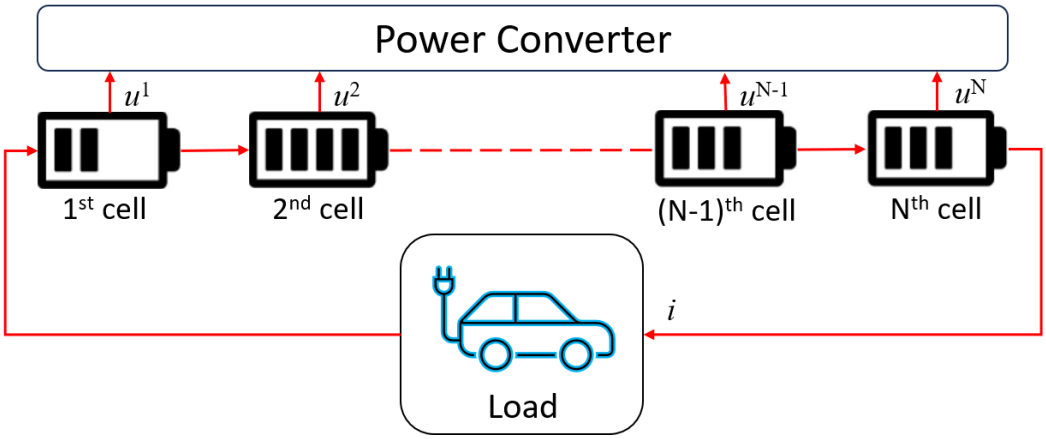

Figure 2. Battery pack configuration for active cell balancing.

For this study, a 5 cell in series pack is assumed resulting in the pack terminal voltage equal to the sum of the cell terminal voltages. For the active cell balancing power converter, an ideal converter with limits on each balancing current and no charge storage is assumed. This power converter also assumes a direct cell-cell topology meaning that charge from any cell can be directed to any other cell with the rate of charge transfer limited by balancing

current limits. These constraints along with the cell model can be formulated as an optimal control problem for MPC at time step k over prediction horizon p as

$$\min_{Z_t, U_t} \quad \sum_{k=1}^{p} J_{mpc}^{k}(Z_t, U_t) \tag{16a}$$

$$\text{s.t.} \quad \zeta_{k+j+1}^{n} = f^{n}(\zeta_{k+j}^{n}, i_{k+j} + u_{k+j}^{n}),$$
$$0 \leq j \leq p-1, 1 \leq n \leq N \tag{16b}$$

$$y_{k+j}^{n} = g^{n}(\zeta_{k+j}^{n}, i_{k+j} + u_{k+j}^{n}),$$
$$1 \leq j \leq p, 1 \leq n \leq N \tag{16c}$$

$$u_{min} \leq u_{k}^{n} \leq u_{max}, \quad 1 \leq n \leq N \tag{16d}$$

$$\underline{y} \leq y_{k+j}^{n}, \quad 1 \leq j \leq p, 1 \leq n \leq N \tag{16e}$$

$$0 = \sum_{i=1}^{N} u_{k}^{n} \tag{16f}$$

The constraints on the optimization problem begin with (16d) to set maximum and minimum limits for each balancing current u_n^k. Next, (16e) states that the cell terminal voltages y_{k+j}^n must be greater than the DVL $\underline{y}$. Finally, (16f) states that there is no charge storage, only transfer with the sum of balancing currents equal to 0.

The cost function chosen for this study is based on minimizing the tracking error between each individual cell to a nominal cell without any imbalance which was studied in [1]. This MPC formulation can be presented mathematically as

$$J_y(u_k) = \sum_{j=1}^{p} (y_{k+j} - y_{k+1}^{0})^T (y_{k+j} - y_{k+1}^{0}) + u_k^T R u_k \tag{17}$$

where y_{k+j} is defined as the vector of cell terminal voltages $y_{k+j} = [y_{k+j}^1, y_{k+j}^2, \ldots y_{k+j}^N]^T$, and R is a positive semi-definitive weighting matrix. The first term penalizes cell voltages that deviate from the nominal cell while the second term penalizes the magnitude of balancing current to reduce resistant heating loses. The nominal cell voltge target is a scalar which leads to needing only the R weighting matrix to tune the cost of the terms.

3.2. RLeMPC for Active Cell Balancing

The MPC controller that solves (16) periodically is then adapted to eMPC by first integrating an event-trigger into the MPC model using the difference between the predicted voltage of a nominal cell and the measured terminal voltage of each cell compared to a calibrate-able threshold $\bar{e}$ as the trigger condition described in (11). When the event is not triggered $\gamma_{ctrl} = 0$, the controller holds the first of the balancing commands of the previous calculated series U_{t_1} instead of using the subsequent elements of U_{t_1}. This modification was used because the future driver power demand is assumed to be unknown and the dynamics of the terminal voltage are relatively slow compared to the controller sample rate of $T_s = 1$ s. This implementation of eMPC was studied to compare to a constant time-triggered MPC baseline and an RL-based event-trigger.

After testing and evaluating the eMPC implementation, an RL agent was developed to replace the trigger conditions. The RL agent was setup as a DQN agent described above in Section II.B. This agent interacts with environment (15) and observes the average and minimum cell voltage, average cell SOC, and pack current demand as state variables as well as the reward (18). Although each cell voltage is considered as the state in (15), only these more general state parameters were used to reduce the number of dimensions of the state for training. During training, the agent will select a random action with probability ϵ that decays exponentially over time and otherwise, will select the action with the greatest value determined by the critic network with parameters ϕ. This action is the event-trigger

for the MCP controller to solve the OCP which then determines a new U_{t_1} and Z_{t_1}. The reward function for each time step was defined as the distance driven over the time step dx subtracted by a flag representing whether or not the event-trigger was set γ_{ctrl} multiplied by a weighting factor ρ together as

$$r = dx - \gamma_{ctrl} * \rho. \tag{18}$$

This reward function is one of the primary design elements for the RLeMPC implementation where the key objectives of maximizing EV range in dx and minimizing event-triggering in γ_{ctrl} are included for the agent to pursue. This reward r is the main feedback signal that the RL agent uses to learn an optimal policy that maximizes r over the episode. The parameter ρ was tuned to achieve the proper scaling between the conflicting objectives to result in the desired behavior of the converged policy. With these RL state and reward formulations, many hyperparamters such as ρ, the structure of the DQN, learning rate, discount factor, and epsilon decay were tuned in a simulation environment while training the agent to find the optimum event trigger policy. Details for training the RLeMPC agent can be found in Algorithm 2 and Fig. 3.

Algorithm 2 RL-based Event-Triggered MPC

procedure RLEMPC(M, T, dt, γ, N)
  Initialize ϕ, $\mathcal{D} \leftarrow \emptyset$, $\phi_t \leftarrow \phi$
  for $j = 0$ to $N - 1$ **do**
    Initialize s_t, Z_{t_1}, U_{t_1}, $k \leftarrow 0$, $u \leftarrow 0$
    while $t <= T$ **do**
      if explore **then**
        Sample a_t from $\{0, 1\}$
      else
        $a_t \leftarrow \arg\max_a Q(s_t, a; \phi)$
      end if
      <Simulate Environment>
      if $\gamma_{ctrl} = 1$ **then**
        $k \leftarrow 0$;
        $(Z_t, U_t) \leftarrow$ Solving OCP (16);
        $u \leftarrow U_t(1)$;
        $U_{t_1} \leftarrow U_t$;
        $Z_{t_1} \leftarrow Z_t$;
      else
        $u \leftarrow U_{t_1}(1)$;
        $k \leftarrow k + 1$;
        if $k <= p$ **then**
          $\hat{\zeta} \leftarrow Z_{t_1}(k)$
        else
          $\hat{\zeta} \leftarrow Z_{t_1}(p)$
        end if
      end if
      $\zeta_{t+1} \leftarrow$ Simulate (13) using u
      $s_{t+1} \leftarrow \zeta_{t+1}$
      $r_t \leftarrow$ (18)
      <End of Environment Simulation>
      Observe r_t and s_{t+1}
      Update $\mathcal{D}$ to include (s_t, a_t, r_t, s_{t+1})
      Sample M experiences from $\mathcal{D}$
      $\phi \leftarrow$ Perform gradient descent on (6)
      if Target Update Condition is True **then**
        $\phi_t \leftarrow \phi$
      end if
      $s_t \leftarrow s_{t+1}$
      $t \leftarrow t + dt$
    end while
  end for
end procedure

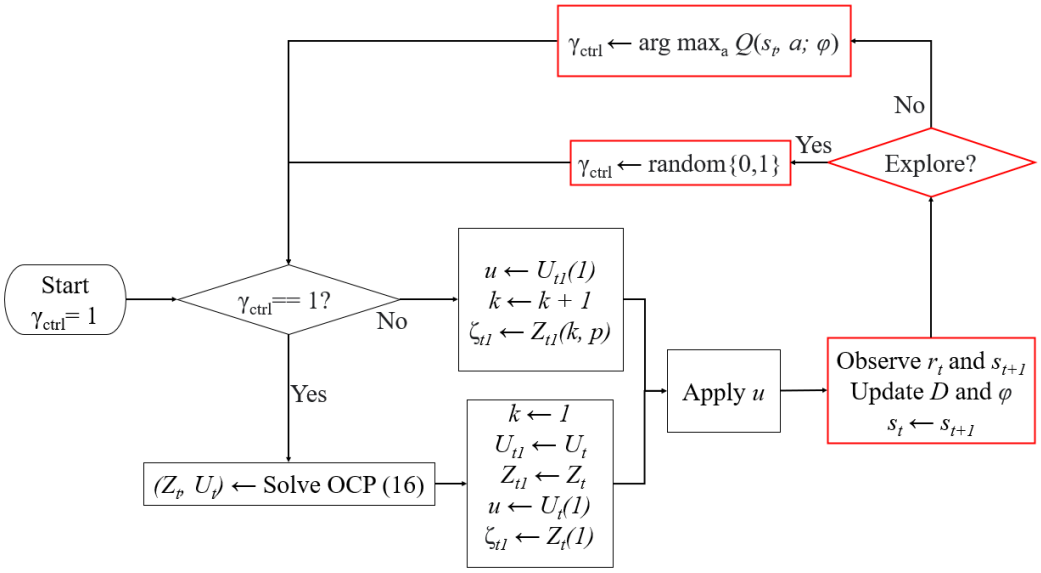

Figure 3. Flowchart for the Reinforcement Learning-Based Event-Triggered MPC (RLeMPC) in Algorithm 2. The red outlines signify the primary changes from the previous eMPC Algorithm 1.

4. Simulation Results and Discussion

4.1. Simulation Environment

Simulations were executed for the training and testing of the active cell balance controls using MATLAB and Simulink with the Reinforcement Learning Toolbox to create and train DQN agents inside of a Simulink environment [48]. The simulations were conducted on a computer with Intel® Core™ i5-6600K processor and 8 GB of RAM. As described above, the system being simulated is a battery pack with 5 cells in series and an ideal power converter that can move charge between any cells constrained by balancing current limits for each cell of ±2 A.

Repeated FTP-72 drive cycle conditions were tested over the sedan EV configuration as used in [1]. The discharge current from the battery pack was scaled from the power demand to assume a larger pack with additional modules in parallel, and a final scaling factor was applied to increase the current draw and reduce simulation time. Starting with all cells fully charged to CVL, the battery was discharged according to the scaled current demand of the vehicle until the first cell reached DVL. The velocity profile and scaled vehicle power demand that was applied to the 5S cell module is shown in Fig 4 where on subsequent cycles, phase 1 of the cycle is repeated for the higher power demands to discharge the battery faster.

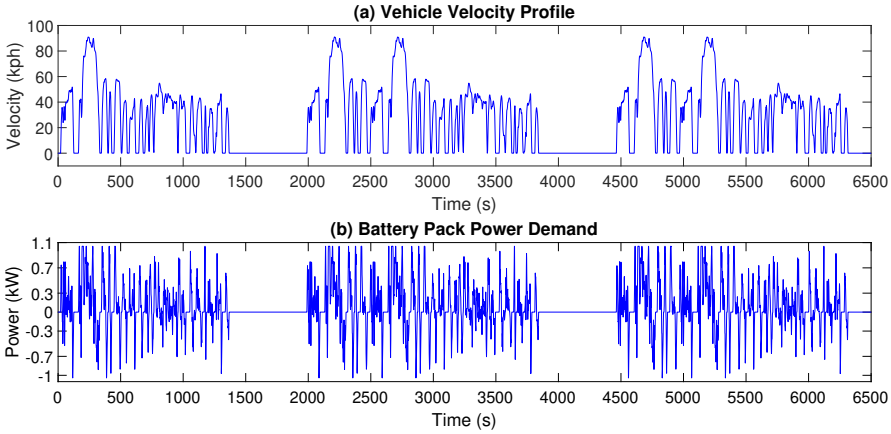

Figure 4. Velocity profile for repeated FTP-72 drive cycles and resulting scaled power requested from 5S cell module applied for the simulations. figure updated

Finally, for all the simulations, a constant imbalance was applied across the C_o^n, R_o^n, C_p^n, and R_p^n cell parameters. To select these parameters, a series of simulations were run that multiplied each of the nominal cell parameters by random factors that were selected from a normal distribution with a mean of 1 and interval between 0.9 and 1.1. From these tests, a set of imbalance factors that resulted in an average active cell balancing range extension benefit for the configuration with the conventional MPC approach was chosen as the constant set of imbalance parameters to use for the rest of the simulations. This was done because the magnitude of the imbalance determines the range extension benefit that can be realized with active cell balancing. With less realizable range extension benefit, the sensitivity of the range extension depending on the control strategy reduces as it changes between MPC, eMPC, and RLeMPC. Future work would include how to generalize these approaches to any distribution of imbalance and quantify the benefit relative to the distribution of cell imbalances, especially for the RL agent which was trained and evaluated using only one set of imbalance parameters for this study.

4.2. Evaluation Criteria

The primary performance metrics for this study are the overall driving range and average event-trigger frequency. The purpose of cell balancing in general is to achieve the maximum energy output of the battery which would translate to maximizing the driving range assuming no auxiliary loads. Moreover, average execution frequency is used as an approximation of the computational load with the goal of minimizing it with RLeMPC. In addition, the magnitude of the balancing currents averaged over the drive cycle is considered to approximate the resistant heating losses and referred to as the balancing effort in the sequel.

4.3. Results with Constant Trigger Period

Before executing simulations with varying trigger frequencies, the MPC weighting matrix R in (17) was re-calibrated to be more robust toward infrequent triggering. For conventional MPC or eMPC with minimal modeling errors, the weighting of R could be decreased to tune the cost optimization toward lower voltage tracking error of the cells to a nominal cell voltage target at the cost of higher balancing currents. However, for this application where future driver power demand is assumed to be unknown, weighting to prioritize aggressively tracking the reference voltage can result in reduced range extension. The unknown future driver power demand becomes a disturbance in the model prediction which increases as the trigger frequency reduces. Furthermore, the prediction horizon may occasionally be much less than the trigger period resulting in prediction error even if power demand was known ahead of time. Because of these unknown dynamics to the model, increasing the cost of the balancing current magnitude through the weighting matrix R can

increase the range performance of the system during infrequent triggering by reducing the response of the controller to a model with large prediction errors.

This dynamic is shown in Fig. 5, where the cost weighting of each balancing current magnitude R^n is set to be equal, scaled appropriately, and tested to determine range extension and balancing effort dependencies. For these tests, the value of R^n was varied for transient cycle discharge tests while having a constant trigger period of 5 s. Increasing the R^n values leads to less balancing effort which demonstrates the effect of R on the MPC cost optimization. For R^n in 800-4000, the range is extended by 0.1% as compared to when R^n is between 10 to 70. Although a small difference, this result demonstrates that with these model assumptions, larger R^n values can result in larger range extension. This result may not be intuitive since range extension is gained with less balancing effort, but this comes from the previously described modeling discrepancies. Once R is very large, the range extension decreases drastically as the balancing currents are greatly reduced limiting the optimal calibration range for R^n. The final R^n calibration was set to 100 to avoid the initial large amounts of balancing effort and also to avoid overly penalizing the balancing effort. Fig. 6 shows the transient cell balancing performance for the final R calibration and a constant trigger period of 5 s.

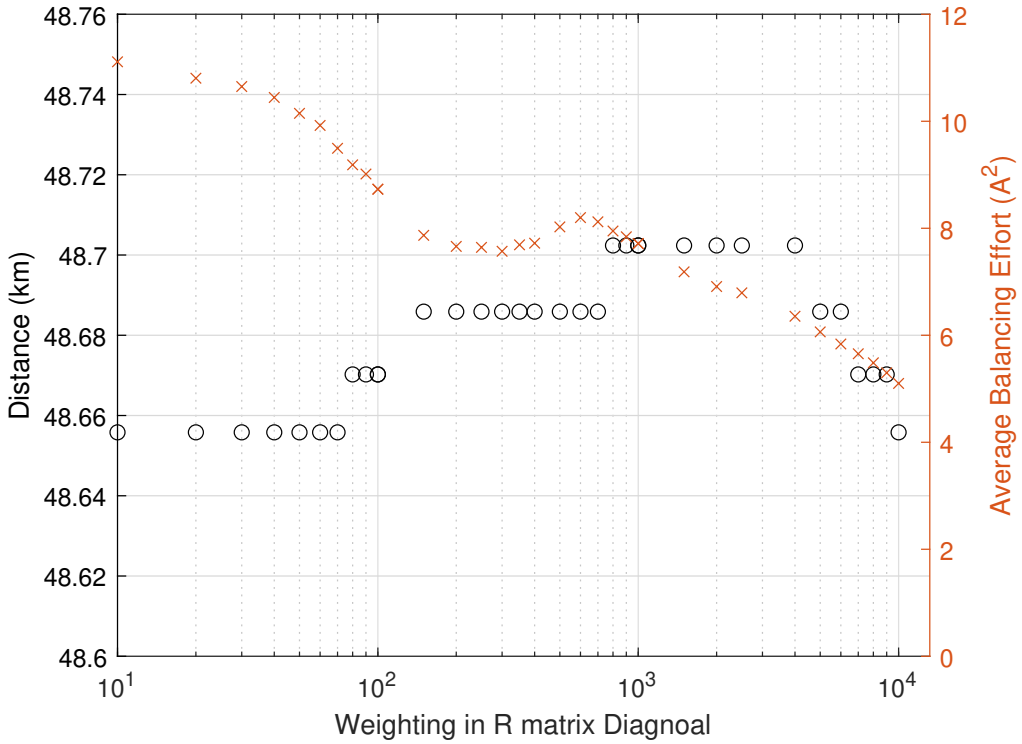

Figure 5. Final EV range and balancing effort results from R weighting matrix sweep with time-triggered MPC with constant trigger period of 5 s.

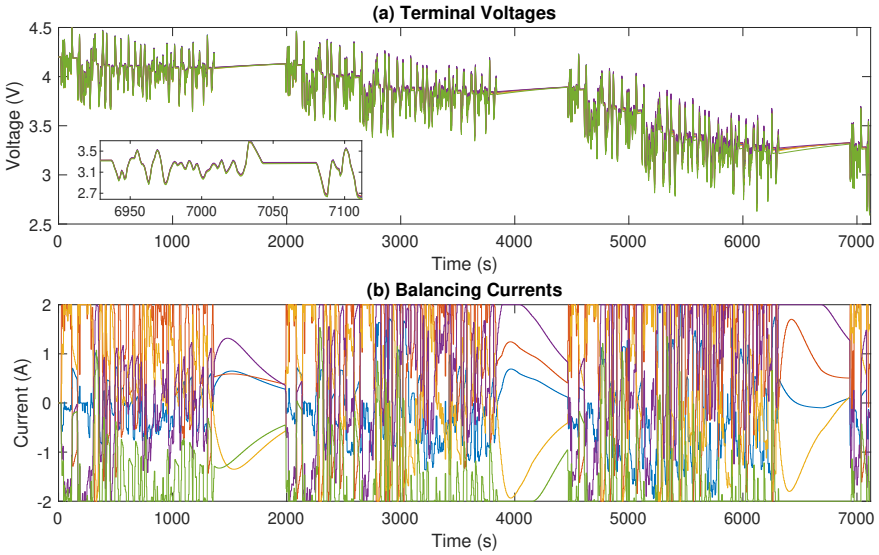

Figure 6. Transient cell voltage and balancing current results for 5 s constant trigger MPC cell balancing. Maximum EV range was achieved. Blue: Cell 1, Red: Cell 2, Yellow: Cell 3, Purple: Cell 4, and Green: Cell 5.

Once the weighting matrix was calibrated, the first study that was completed was varying a constant trigger frequency. These tests were simulated to understand as a baseline, without any event-trigger, the range extension and balancing current magnitudes with only varying the trigger frequency of the MPC controller. For this purpose, the trigger frequency was set to a constant value starting from 1Hz and decreased between simulations until 0 triggers occurred. This trigger frequency range demonstrates the maximum and minimum driving ranges achievable. For these tests, the prediction horizon remained as 5 s with only the first element of the control sequence U_t being applied until another trigger occurred. Notably, the time period between triggers can be greater than the prediction horizon as described in Section III.B.

The transient cycle range extension results plotted in Fig. 7 indicate that the maximum range of 48.66-48.67 km can be achieved with a constant trigger period of 1 to 700 s or frequency of 1Hz to 1 mHz with varying balancing effort as the trigger period changed. Noticeably, two discrete ranges emerge with the higher trigger frequency tests from 1 Hz down to 1 mHz resulting around this maximum range and tests with frequencies below 1 mHz ending around a minimum range of 46.23-46.24 km. Fig. 8 shows effective cell balancing with a constant trigger period of 1000 s and can be compared to Fig. 6 to notice the much less busy balancing current which delivered nearly the same final driving distance. The discrete driving range levels are attributable to the current scaling that was applied to the transient cycle simulations which leads to these discrete windows emerging for when the DVL is reached.

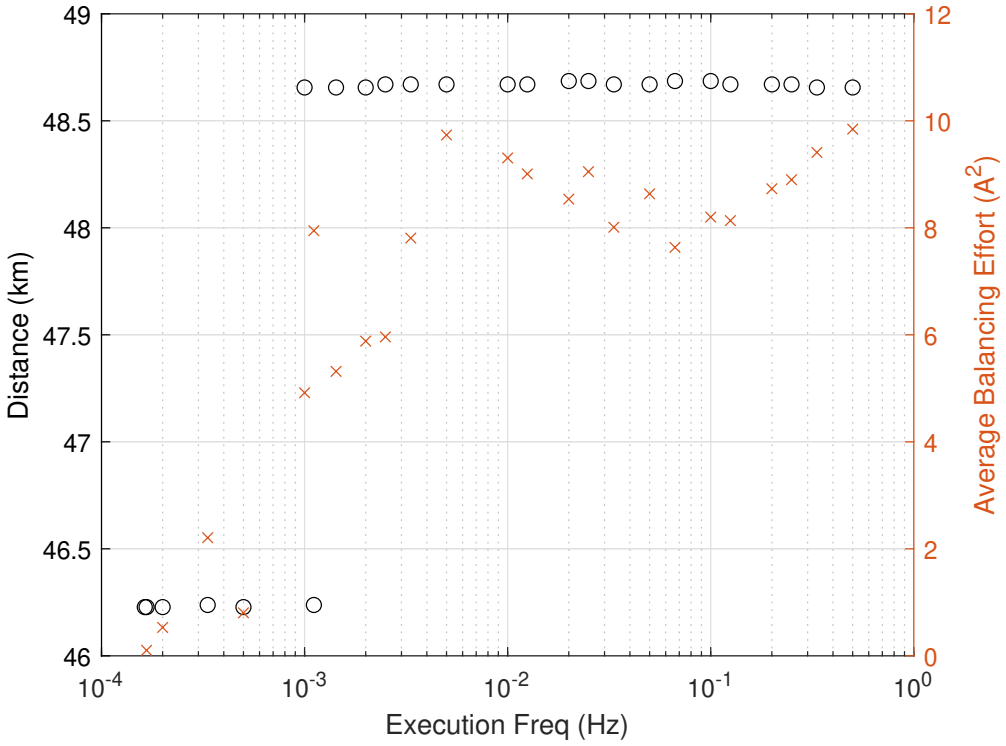

Figure 7. EV range and balancing effort results from varying the constant trigger period of conventional MPC.

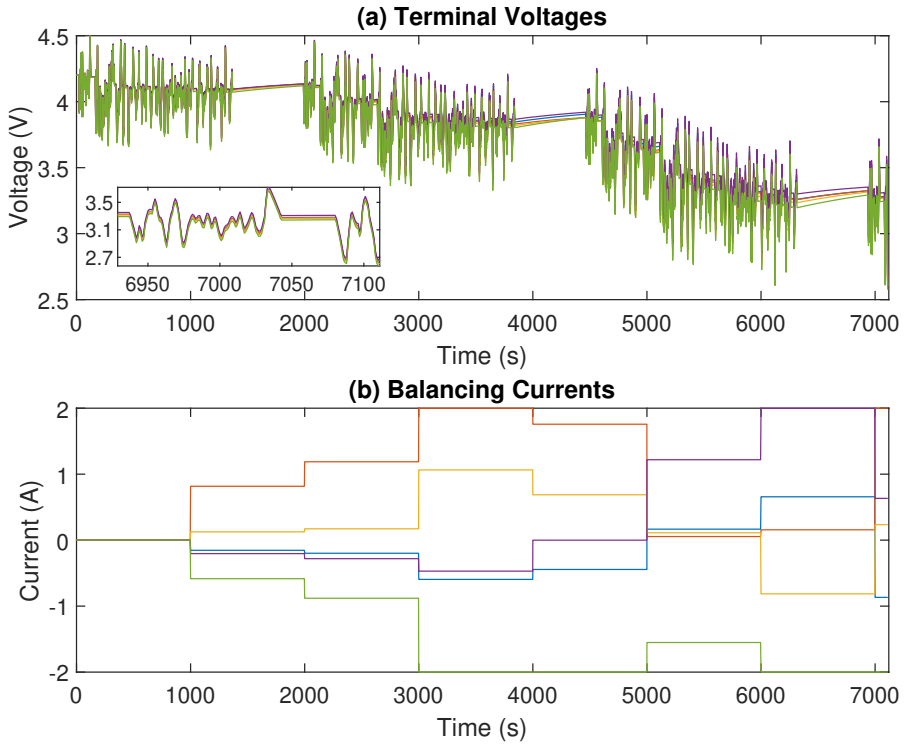

Figure 8. Transient cell voltage and balancing current results for 1000 s constant trigger MPC cell balancing. The infrequent event-triggers are clearly visible with the steps in the balancing currents. Near maximum EV range is still achieved with much reduced triggering. Blue: Cell 1, Red: Cell 2, Yellow: Cell 3, Purple: Cell 4, and Green: Cell 5.

To expand on this, Fig. 9 shows the vehicle velocity profile along with the transient pack current demand and minimum cell voltage for a simulation that did not reach DVL during the time window where other simulations with poor cell balancing did reach DVL. This occurs on the third repeated cycle where DVL is reached at a pack SOC of 33% and current demand of 280-380 A. The simulations with higher trigger frequencies continued for a fourth cycle until DVL was reached at a pack SOC of 29% and close to 400 A cell current demand. Cell terminal voltages decreased significantly during these high current discharges due to the large internal resistance of the cell. This current scaling along with the nonlinear OCV creates these discrete final driving ranges such that the first current peak where DVL is reached may be overcome with cell balancing but the second current peak where the rest of the simulations reach DVL cannot be overcome. The second current peak cannot be overcome even with perfectly balanced cells and a significant 29% SOC remaining because of the large current demands. This effect could be smoothed out with more realistic cell currents at the cost of simulation time. However for this study, especially for testing RLeMPC, simulation times had to be low to train the RL agent in a reasonable amount of time. These results are sufficient for initial concept demonstration to test if RLeMPC can determine the optimal eMPC trigger policy to overcome the first high power window. plot eMPC transient results

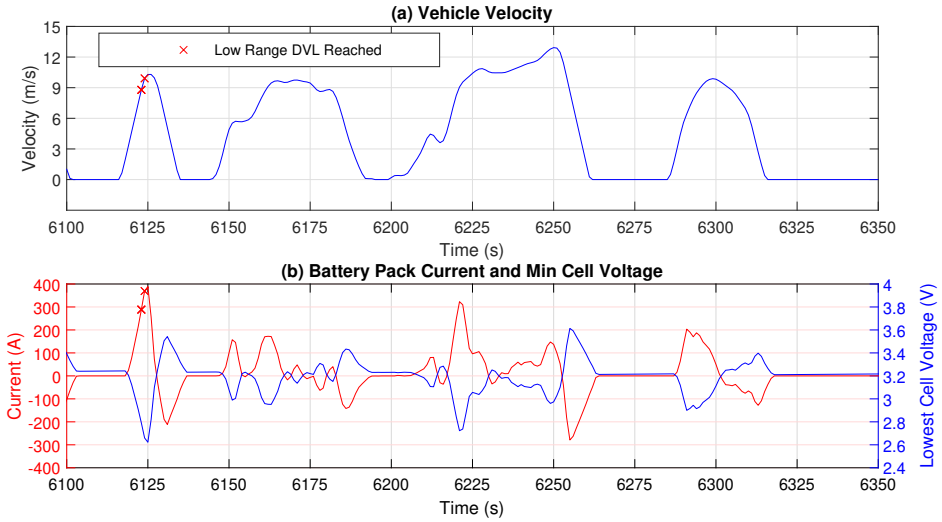

Figure 9. Constant trigger velocity and current profiles from maximum EV range tests in the range where DVL is reached for minimum EV range tests. The red x's mark the vehicle velocities and large currents where DVL is reached for those tests. Notably, after overcoming the peak current demand, the vehicle can travel much further with lower currents and higher cell voltages. fig updated

4.4. Results with Threshold Based eMPC

For eMPC, simulations with varying the error threshold were executed to understand if an eMPC approach could outperform the constant trigger frequency MPC controller in terms of range extension, average trigger frequency, and balancing effort. Fig. 10 shows as the error threshold increased, the trigger frequency decreased approximately exponentially until saturating at 0-1 trigger per test. Between an error threshold of 1 and 1.5 V, the exponential relationship begins to break as the cell balancing fails to avoid DVL during the first very high current peak as illustrated in the large step of the driving range around 10 mHz with Fig. 11. Compared to the constant trigger frequency results, eMPC did not achieve as much range extension as MPC at reduced average triggering frequencies. For example, most of the frequency range between 10 mHz and 1 mHz reached DVL at a driving range of 46.24 km for eMPC while for constant trigger frequency MPC at this frequency range, DVL was reached at 48.66 km. Additionally, the average balancing effort is higher on average and more variable for eMPC. Overall, constant trigger frequency MPC

performs better than eMPC over the evaluation criteria for this application and trigger condition highlighting the challenge of implementing an optimal eMPC trigger condition.

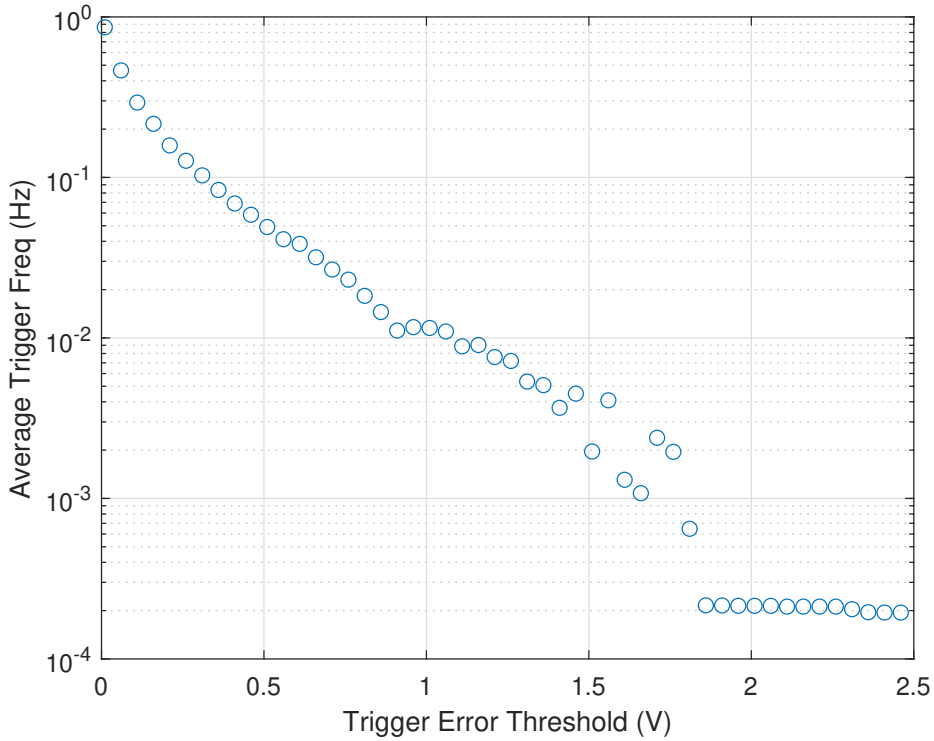

Figure 10. Average execution frequency as a function of the error threshold for eMPC calibration.

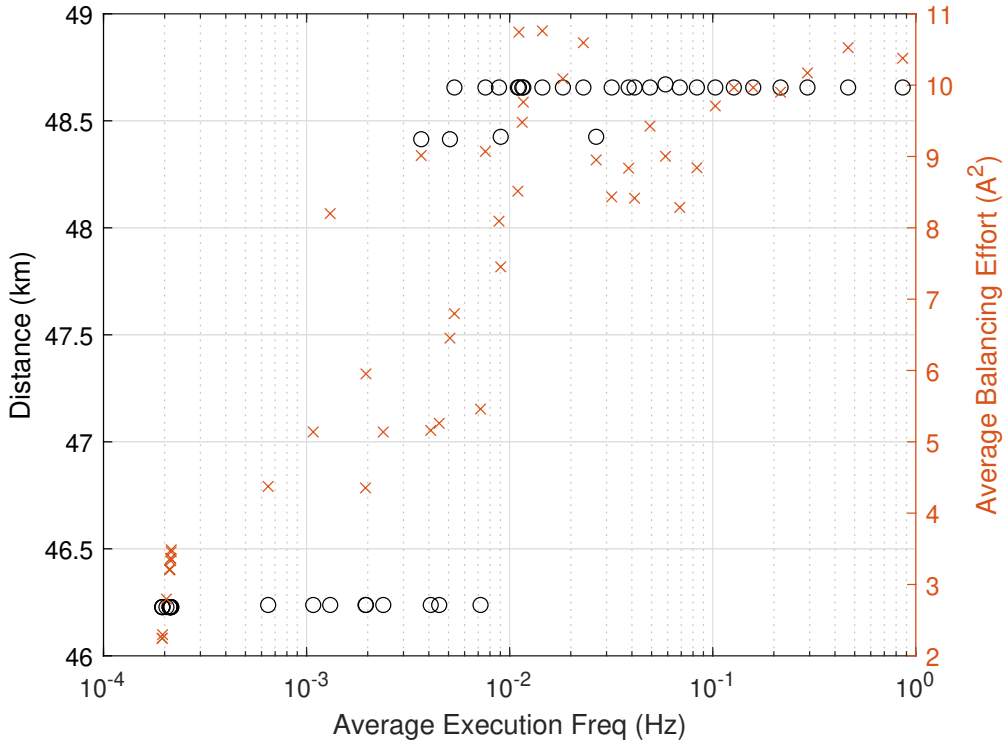

Figure 11. EV range and balancing effort depending on the average execution period for eMPC. Notably, the switch from high EV range to low EV range occurs at a higher average execution frequency than constant triggering MPC.

Transient results for eMCP with the lowest average event-trigger frequency that achieved 48.66 km are plotted in Fig. 12. In this example, the benefit of eMPC is demonstrated in the lack of event-triggers during the standstill portions between repeated cycles when triggering is not required. This control strategy can be effective when the load is 0 or constant, but during the transient portions of the test, the actual cell voltages are much more transient relative to the nominal voltage target computed when the OCP is solved causing excessive triggering. Changing the target voltage to the average actual cell voltage instead of a nominal predicted cell target may be an improvement on the event-trigger policy to account for the modeling errors arising from unpredictable driver power request.

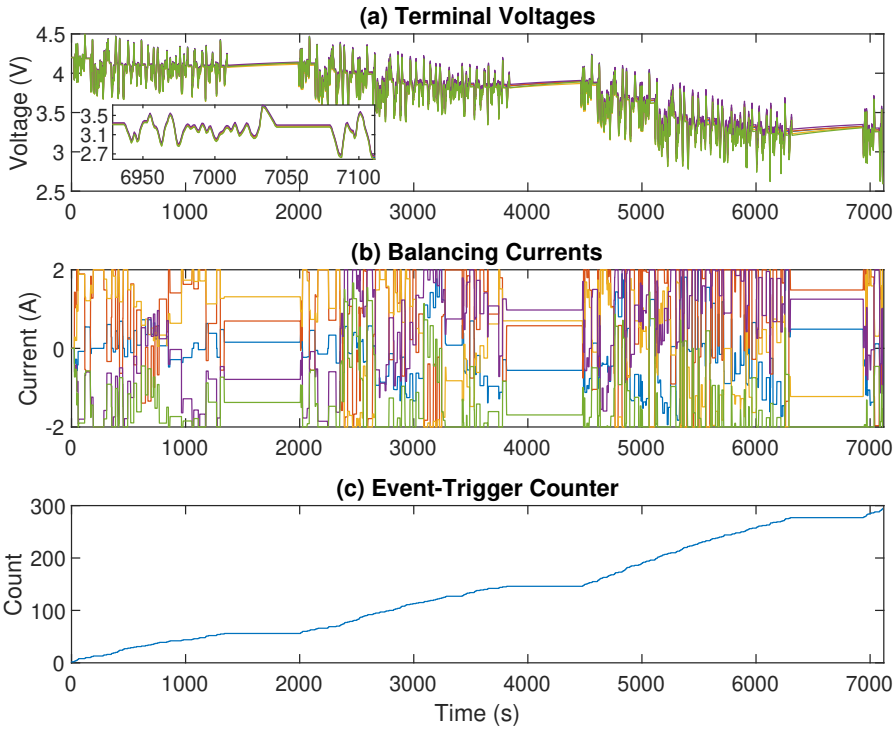

Figure 12. Transient cell voltage, balancing current, and event-trigger results for eMPC with $\bar{e} = 1.31$ V. Error threshold $\bar{e}$ is set to achieve high EV range with minimal event-triggers. Blue: Cell 1, Red: Cell 2, Yellow: Cell 3, Purple: Cell 4, and Green: Cell 5.

4.5. Results with RLeMPC

Many challenges were encountered while training the RLeMPC agent. For this problem, only a minimal amount of triggers is required to achieve the maximum reward, and any amount of triggers below the minimum results in very little range extension benefit. This feature of the dynamics leads to the RL agent learning a sub-optimal policy of triggering very infrequently or the training can diverge to triggering excessively. Because of the penalty of triggering in the reward function, the RL agent can learn a policy of not triggering at all which would be the optimal policy if the first current peak could not be overcome with cell balancing. Depending on how the RL agent is trained, the agent can erroneously learn this as the optimal policy even after having experienced the larger driving ranges that are achievable. While training the agent, this local optimum became very difficult to avoid even with a low weighting factor ρ applied to the trigger action in the reward function.

Two important training parameters that were explored to avoid this local optimum were the discount factor γ and the ϵ-decay exploration method. To start with γ, typical values like 0.9 and 0.99 overly discounted the future delayed range extension reward from cell balancing thousands of steps in advance. To account for these delayed rewards, γ in the range of 0.999-0.9999 was required, but such a high value for γ required more training to accurately learn which actions lead to the delayed rewards. Extended training time or a more refined approach to addressing the delayed rewards in this problem may be required to learn the optimal policy. This feature differentiates the active cell balancing problem from other problems with less delayed rewards such as autonomous vehicle path following where RLeMPC was successfully applied in [15].

The ϵ-decay method for training was another major challenge for this problem. Typically, ϵ is initialized around 1 and decayed to a minimum value to explore the environment before transitioning to exploiting the environment to maximize G. For this problem, the maximum final range can be achieved with very little triggering, so with high ϵ, and 50%

probability of choosing to trigger for a random action, the agent first learned the penalty from triggering because it always achieved the maximum range at the beginning of the training. As ϵ decayed, the agent transitioned from random actions to following the actions with greater learned Q value from the DQN which was initially to not trigger at all. Eventually, ϵ decayed enough to where not enough triggering occurred from random actions to achieve the maximum range, and during this transition, it was very difficult for the agent to learn that triggering more can lead to much greater range extension. Part of this could come from the fact that the agent could not learn any distinction in driving range extension between triggering and no triggering at the beginning of the training. Next, once it did not trigger enough to overcome the first high current peak, it could not learn quickly enough that triggering could lead to larger range extension reward before falling into the local optimum of not triggering at all with no driving range extension.

This example of training plays out in Fig. 13(a) where the reward correlates with ϵ from the decrease in trigger frequency until around 400 episodes, after which triggering was not frequent enough to overcome the first high current peak. The switching observed in the episode reward was from going between the high and low final driving ranges with large reward weighting applied to the driving range extension. After this transition, the RL agent settled into a policy of not triggering at all to maximize the reward with no driving range extension.

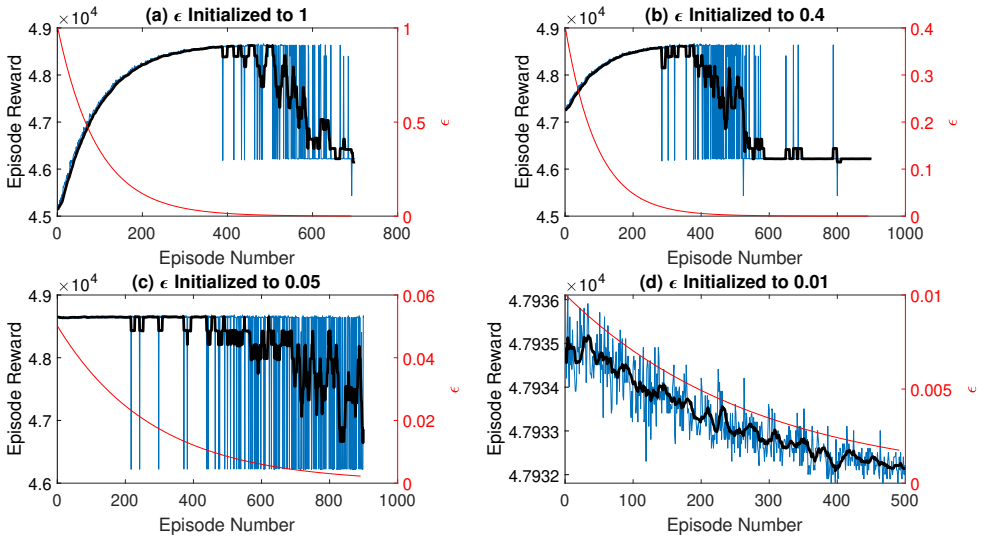

Figure 13. Reward and ϵ-decay during RLeMPC training. Blue: raw episode reward; Black: moving average of episode reward. (a) is with ϵ initalized to 1 and agent learns to reduce triggering even after experiencing EV range degradation. (b) and (c) start with lower initial ϵ and also learn to minimize triggering at the expense of EV range. (d) begins with ϵ of 1% and learns to increase triggering even while consistently achieving maximum range.

To attempt to overcome this challenge of learning a sub-optimal policy during the exploration phase of the training, ϵ was initialized to much lower values for training. These values were chosen from the maximum episode length of around 7000 steps with only two actions available per step which still results in a significant number of exploratory actions. The MPC and eMPC results show around 100-1000 s per trigger or 7-70 triggers per episode is the minimum amount of triggers required to achieve large driving range extension. This new ϵ-decay tuning along with random initial weights and biases for the DQN was chosen to try to initialize the training right on the step of low to high driving range extension so that the agent experiences the range extension difference between triggering and not triggering right away.

Attempts at this ϵ-decay tuning have not improved the training performance. First, when ϵ is not low enough, the agent learns to not trigger falling into the local optimum of 0 triggers per episode as shown in Fig. 13(b) and (c). For these training sets, ϵ was initialized to 40% and 5%, and both resulted in the same policy of no triggering to maximize reward when no driving range extension is achieved. From those training sets, ϵ of 1-5% appears to be the best value to begin the training since in that range is where the switching between high and low driving distance ranges begins to be observed. However, initializing ϵ to 1% shown in Fig. 13(d) resulted in the training diverging and the reward decreasing with the maximum distance was always achieved but the RL agent learning to trigger more.

Next steps that are planned for improving upon the results presented include focused tuning of γ, ϵ-decay, and the reward weighting ρ around the driving range transition trigger frequencies. The expectation is that the data efficiency of the training can be increased with focused exploration in the key trigger frequency range. The first step is to find a proper initial value of ϵ as well as a more effective ϵ-decay rate. As mentioned, an initial ϵ value of 1-5% should be appropriate to place the initial trigger frequency around where the step of driving range occurs, but slowing down the ϵ-decay rate may help give the agent more time to explore in the key ϵ range. Additionally, varying the initial DQN weights and reward weighting may help to avoid the diverging behavior observed in Fig. 13(d). Finally, tuning γ in this more effective exploration range should be beneficial to ensure delayed rewards are weighted properly to attribute the range extension benefit to the earlier triggering.

Overall, the analysis from tuning this DQN agent for active cell balancing can provide guidance for tuning γ, ϵ-decay, and the reward function for a particular problem when training for tens of thousands of episodes is not a viable option. These next steps identified will be included in future work to report on their effectiveness with improving the performance of training the RL agent. One glimmer of the potential comes from the switching observed in rewards at low ϵ values in Fig. 13. The high values in that switching represent episodes where the RL agent achieved the larger range extension with very little triggering from random actions. For example Fig. 14 shows the cell balancing and triggering occurring during one of those episodes where only 7 random triggers resulted in a final driving range of 48.66 km. This performance is on par with the constant infrequent MPC triggering approach showing improvement is available if the RL agent can be trained to learn it. Finally, adjusting the current scaling of the simulation may also help shape the Q function to be more variable, easier to learn, and more reflective of the actual cell balancing problem.

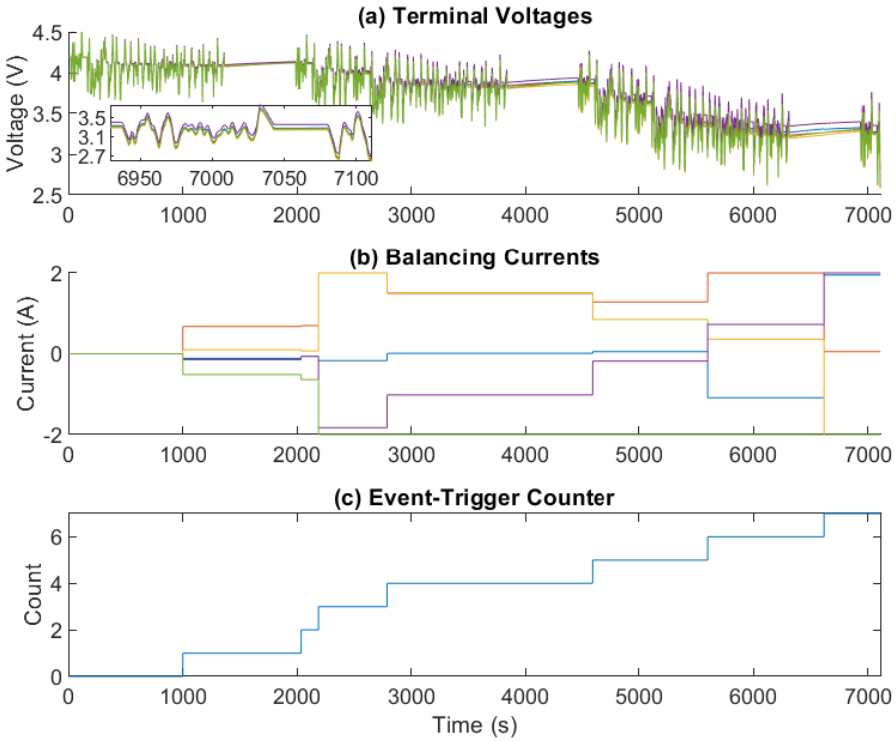

Figure 14. RLeMPC training episode where learned policy is to not trigger at all, but random off-policy triggers achieve large range extension with very infrequent event-triggers. Blue: Cell 1, Red: Cell 2, Yellow: Cell 3, Purple: Cell 4, and Green: Cell 5.

5. Conclusions

Three model predictive control (MPC) strategies, namely time-triggered MPC, event-triggered MPC (eMPC), and reinforcement learning-based MPC (RLeMPC), for active cell balancing, were formulated and tested in a simulation environment to reduce computational requirements relative to a baseline MPC controller while maintaining the EV range extension benefit. MPC with reduced constant period triggering and eMPC control methods demonstrated significant computational load reduction with an average trigger period increase to 1000 s and 187 s respectively compared to the 1 s trigger period of the baseline MPC. Only a negligible decrease of EV range extension of 0.03% from the maximum achievable range was the penalty for this significant decrease in throughput. From scaling of the current demand for more efficient simulation, the discharge voltage limit was met at very high cell current demands between 280-380 A and 33% state of charge remaining in the battery pack shaping the EV range extension results. Challenges of training the RLeMPC agent were presented such as the discrete and delayed range extension rewards as well as an ineffective exploration method. Overall, the converged RLeMPC policy was very sensitive to training which will be further improved with hyper-parameter tuning, but occasional training episodes were promising with greater than 1000 s average trigger period. Future steps to address these challenges, to learn more optimal event-trigger policies, and to improve the robustness of the proposed approach, were discussed and are planned as future work. Real-world implementation in microcontroller is another future work direction.

Author Contributions: Conceptualization, D.F. and J.C.; methodology, J.C; software, D.F.; validation, D.F., J.C. and G.X.; formal analysis, D.F..; data curation, D.F.; writing—original draft preparation, D.F.; writing—review and editing, D.F., J.C., and G.X; visualization, D.F.; supervision, J.C.; project administration, J.C.; funding acquisition, J.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by Oakland University through SECS Faculty Startup Fund and URC Faculty Research Fellowship and in party by National Science Foundation through Award #2237317.

Data Availability Statement: The data is available upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. J. Chen, A. Behal, and C. Li, "Active battery cell balancing by real time model predictive control for extending electric vehicle driving range," *IEEE Transactions on Automation Science and Engineering*, accepted June 2023.
2. Daowd, M., Omar, N., Van Den Bossche, P. & Van Mierlo, J. Passive and active battery balancing comparison based on MATLAB simulation. *2011 IEEE Vehicle Power And Propulsion Conference*. pp. 1-7 (2011)
3. M. Preindl, "A battery balancing auxiliary power module with predictive control for electrified transportation," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6552–6559, 2017.
4. J. Liu, Y. Chen, and H. K. Fathy, "Nonlinear model-predictive optimal control of an active cell-to-cell lithiumion battery pack balancing circuit," *IFAC PapersOnLine*, vol. 50, no. 1, pp. 14 483–14 488, 2017.
5. P. Tøndel, T. A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit mpc solutions," *Automatica*, vol. 39, no. 3, pp. 489–497, 2003.
6. Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.
7. R. Badawi and J. Chen, "Performance evaluation of event-triggered model predictive control for boost converter," in *2022 IEEE Vehicle Power and Propulsion Conference*, Merced, CA, November 1–4, 2022.
8. Karnehm, D., Bliemetsrieder, W., Pohlmann, S. & Neve, A. Controlling Algorithm of Reconfigurable Battery for State of Charge Balancing using Amortized Q-Learning. (Preprints,2024)
9. Razmjooei, H., Palli, G., Abdi, E., Terzo, M. & Strano, S. Design and experimental validation of an adaptive fast-finite-time observer on uncertain electro-hydraulic systems. *Control Engineering Practice*. **131** pp. 105391 (2023)
10. H. Li and Y. Shi, "Event-triggered robust model predictive control of continuous-time nonlinear systems," *Automatica*, vol. 50, no. 5, pp. 1507–1513, 2014.
11. F. D. Brunner, W. Heemels, and F. Allgöwer, "Robust event-triggered MPC with guaranteed asymptotic bound and average sampling rate," *IEEE Trans. Autom. Control*, vol. 62, no. 11, pp. 5694–5709, 2017.
12. Z. Zhou, C. Rother, and J. Chen, "Event-triggered model predictive control for autonomous vehicle path tracking: Validation using CARLA simulator," *IEEE Transactions on Intelligent Vehicles*, accepted for publication April 2023.
13. J. Yoo and K. H. Johansson, "Event-triggered model predictive control with a statistical learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 4, pp. 2571–2581, 2021.
14. R. Badawi and J. Chen, "Enhancing enumeration-based model predictive control for dc-dc boost converter with event-triggered control," in *2022 European Control Conference*, London, UK, July 12–15, 2022.
15. J. Chen, X. Meng, and Z. Li, "Reinforcement learning-based event-triggered model predictive control for autonomous vehicle path following," in *American Control Conf.*, Atlanta, GA, June 8–10, 2022.
16. F. Dang, D. Chen, J. Chen, and Z. Li, "Event-triggered model predictive control with deep reinforcement learning," *IEEE Transactions on Intelligent Vehicles*, accepted for Publication, October 2023.
17. D. Baumann, J.-J. Zhu, G. Martius, and S. Trimpe, "Deep reinforcement learning for event-triggered control," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 943–950.
18. A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep reinforcement learning for wireless sensor scheduling in cyber–physical systems," *Automatica*, vol. 113, p. 108759, 2020.
19. J. Chen and Z. Zhou, "Battery cell imbalance and electric vehicles range: Correlation and NMPC-based balancing control," in *2023 IEEE International Conference on Electro Information Technology*, Romeoville, IL, May 18–20, 2023.
20. M. Dubarry, N. Vuillaume, and B. Y. Liaw, "Origins and accommodation of cell variations in li-ion battery pack modeling," *International Journal of Energy Research*, vol. 34, no. 2, pp. 216–231, 2010.
21. J. Chen, Z. Zhou, Z. Zhou, X. Wang, and B. Liaw, "Impact of battery cell imbalance on electric vehicle range," *Green Energy and Intelligent Transportation*, vol. 1, no. 3, pp. 1–8, December 2022.
22. F. S. J. Hoekstra, H. J. Bergveld, and M. Donkers, "Range maximisation of electric vehicles through active cell balancing using reachability analysis," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 1567–1572.
23. Y. Shang, N. Cui, and C. Zhang, "An optimized any-cell-to-any-cell equalizer based on coupled half-bridge converters for series-connected battery strings," *IEEE Transactions on Power Electronics*, vol. 34, no. 9, pp. 8831–8841, 2018.
24. C. Wang, G. Yin, F. Lin, M. P. Polis, C. Zhang, J. Jiang *et al.*, "Balanced control strategies for interconnected heterogeneous battery systems," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 1, pp. 189–199, 2015.
25. M. Evzelman, M. M. U. Rehman, K. Hathaway, R. Zane, D. Costinett, and D. Maksimovic, "Active balancing system for electric vehicles with incorporated low-voltage bus," *IEEE Transactions on Power Electronics*, vol. 31, no. 11, pp. 7887–7895, 2015.

26. J. Xu, B. Cao, S. Li, B. Wang, and B. Ning, "A hybrid criterion based balancing strategy for battery energy storage systems," *Energy Procedia*, vol. 103, pp. 225–230, 2016.
27. Z. Gao, C. Chin, W. Toh, J. Chiew, and J. Jia, "State-of-charge estimation and active cell pack balancing design of lithium battery power system for smart electric vehicle," *Journal of Advanced Transportation*, vol. 2017, no. Article ID 6510747, 2017.
28. S. Narayanaswamy, S. Park, S. Steinhorst, and S. Chakraborty, "Multi-pattern active cell balancing architecture and equalization strategy for battery packs," in *Proc. of the International Symposium on Low Power Electronics and Design*, Seattle, WA, July 23–25, 2018, pp. 1–6.
29. M. Kauer, S. Narayanaswamy, S. Steinhorst, M. Lukasiewycz, and S. Chakraborty, "Many-to-many active cell balancing strategy design," in *The 20th Asia and South Pacific Design Automation Conference*, Chiba, Japan, 2015, pp. 267–272.
30. M. Einhorn, W. Roessler, and J. Fleig, "Improved performance of serially connected li-ion batteries with active cell balancing in electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 6, pp. 2448–2457, 2011.
31. F. S. Hoekstra, L. W. Ribelles, H. J. Bergveld, and M. Donkers, "Real-time range maximisation of electric vehicles through active cell balancing using model-predictive control," in *2020 American Control Conference*, Denver, CO, July 1–3, 2020, pp. 2219–2224.
32. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An introduction*. MIT Press, 2018.
33. C. J. C. H. Watkins, "Learning from delayed rewards," 1989.
34. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013.
35. Preindl, M. A battery balancing auxiliary power module with predictive control for electrified transportation. *IEEE Transactions On Industrial Electronics*. **65**, 6552-6559 (2017)
36. Pinto, C., Barreras, J., Schaltz, E. & Araujo, R. Evaluation of advanced control for li-ion battery balancing systems using convex optimization. *IEEE Transactions On Sustainable Energy*. **7**, 1703-1717 (2016)
37. McCurlie, L., Preindl, M. & Emadi, A. Fast model predictive control for redistributive lithium-ion battery balancing. *IEEE Transactions On Industrial Electronics*. **64**, 1350-1357 (2016)
38. Altaf, F., Egardt, B. & Mårdh, L. Load management of modular battery using model predictive control: Thermal and state-of-charge balancing. *IEEE Transactions On Control Systems Technology*. **25**, 47-62 (2016)
39. Liu, J., Chen, Y. & Fathy, H. Nonlinear model-predictive optimal control of an active cell-to-cell lithium-ion battery pack balancing circuit. *IFAC-PapersOnLine*. **50**, 14483-14488 (2017)
40. H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.
41. J. Chen and Z. Yi, "Comparison of event-triggered model predictive control for autonomous vehicle path tracking," in *IEEE Conf. Control Technology and Applications*, San Diego, CA, August 8–11, 2021.
42. Mestrallet, F., Kerachev, L., Crebier, J. & Collet, A. Multiphase interleaved converter for lithium battery active balancing. *IEEE Transactions On Power Electronics*. **29**, 2874-2881 (2013)
43. Maharjan, L., Inoue, S., Akagi, H. & Asakura, J. State-of-charge (SOC)-balancing control of a battery energy storage system based on a cascade PWM converter. *IEEE Transactions On Power Electronics*. **24**, 1628-1636 (2009)
44. Z. Pei, X. Zhao, H. Yuan, Z. Peng, and L. Wu, "An equivalent circuit model for lithium battery of electric vehicle considering self-healing characteristic," *Journal of Control Science and Engineering*, vol. 2018, 2018.
45. J. Wehbe and N. Karami, "Battery equivalent circuits and brief summary of components value determination of lithium ion: A review," in *2015 Third International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE)*, Beirut, Lebanon, 2015, pp. 45–49.
46. Yu, L., Xia, Y. & Sun, Z. Robust event-triggered model predictive control for constrained linear continuous system. *International Journal Of Robust And Nonlinear Control*. **29** (2018,12)
47. Li, H. & Shi, Y. Event-triggered robust model predictive control of continuous-time nonlinear systems. *Automatica*. **50**, 1507-1513 (2014), https://www.sciencedirect.com/science/article/pii/S0005109814001071
48. The MathWorks Inc., "Deep Q-Network (DQN) Agents," Natick, Massachusetts, United States, accessed: 2024-01-29. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/dqn-agents.html#d126e7212
49. Shibata, K., Jimbo, T. & Matsubara, T. Deep reinforcement learning of event-triggered communication and consensus-based control for distributed cooperative transport. *Robotics And Autonomous Systems*. **159** pp. 104307 (2023)
50. Abbasimoshaei, A., Chinnakkonda Ravi, A. & Kern, T. Development of a new control system for a rehabilitation robot using electrical impedance tomography and artificial intelligence. *Biomimetics*. **8**, 420 (2023)
51. Zhang, Y., Huang, Y., Chen, Z., Li, G. & Liu, Y. A novel learning-based model predictive control strategy for plug-in hybrid electric vehicle. *IEEE Transactions On Transportation Electrification*. **8**, 23-35 (2021)
52. Rostam, M. & Abbasi, A. A framework for identifying the appropriate quantitative indicators to objectively optimize the building energy consumption considering sustainability and resilience aspects. *Journal Of Building Engineering*. **44** pp. 102974 (2021)