Reinforcement Learning Building Control: An Online Approach with Guided Exploration using Surrogate Models

Sourav Dey[1]
Department of Civil, Environmental and Architectural Engineering
University of Colorado Boulder,
Boulder, Colorado 80309-0428, USA
email: sourav.dey@colorado.edu

Gregor P. Henze
Department of Civil, Environmental and Architectural Engineering
University of Colorado Boulder,
Boulder, Colorado 80309-0428, USA
email: gregor.henze@colorado.edu

The incorporation of emerging technologies, including solar photovoltaics, electric vehicles, battery energy storage, smart devices, internet-of-things (IoT) devices, and sensors in buildings, desirable control objectives are becoming increasingly complex, calling for advanced controls approaches. Reinforcement learning (RL) is a powerful method for this. RL can adapt and learn from environmental interaction, but it can take a long time to learn and can be unstable initially due to limited environmental knowledge. In our research, we propose an online RL approach for buildings that uses data-driven surrogate models to guide the RL agent during its early training. This helps the controller learn faster and more stably than the traditional direct plug-and-learn online learning approach. In this research, we propose an online approach in buildings with RL where, with the help of data-driven surrogate models, the RL agent is guided during its early exploratory training stage, aiding the controller to learn a near-optimal policy faster and exhibiting more stable training progress than a traditional direct plug-and-learn online learning RL approach. The agents are assisted in their learning and action with information gained from the surrogate models generating multiple artificial trajectories starting from the current state. The research presented an exploration of various surrogate model-assisted training methods and revealed that models focusing on artificial trajectories around rule-based controls yielded the most stable performance. In contrast, models employing random exploration with a one-step look-ahead approach demonstrated superior overall performance.

Keywords: energy systems, artificial intelligence, control system, reinforcement learning

Nomenclature

EV = Electric vehicle
PID = Proportional integrative derivative
RBC = Rule based control
RLC = Reinforcement learning control
MPC = Model predictive control
IAQ = Indoor air quality
IoT = Internet of things
ML = Machine learning
IAQ = Planck constant
MDP = Markov decision process
NLP = Natural language processing
ANN = Artificial neural network
DNN = Deep neural network
ACTB = Advanced Controls Test Bed
ELM = Extreme learning machine
RTU = Roof top unit
A3C = Asynchronous Advantage Actor Critic
LTI = Linear time invariant
PV = Photovoltaic
DQN = Deep Q-Network
SAC = Soft actor-critic
KPI = Key performance indicator
AHU = Air handling unit
DDPG = Deep-deterministic-policy-gradient
HVAC = Heating, ventilation and air conditioning
AHU = Air handling unit

[1]Corresponding Author.
Version 1.18, January 31, 2026

1 Introduction

Effective building control is becoming increasingly important in the management and operation of building energy systems. Its key functions include ensuring comfortable and healthy indoor environments and minimizing operational energy costs. Globally, building operations contribute to approximately 30% of primary energy usage and 28% of greenhouse gas emissions. In the United States, these figures are even higher, with buildings accounting for 39% of primary energy use and 30% of greenhouse gas emissions [1–3]. Traditionally, building controllers were designed to balance energy consumption while maintaining healthy and comfortable indoor conditions. However, the emergence of new technologies like Internet of Things (IoT) devices, sensors, on-site renewable energy sources, energy storage, and electric vehicle (EV) charging stations necessitates advanced control systems. These novel control systems are expected to manage multiple objectives, such as energy efficiency, grid requirements, indoor air quality (IAQ), occupant preferences, and others, while also reducing energy use and carbon emissions and minimizing thermal discomfort [4,5].

There are several control approaches that have been utilized in the domain of building controls. The most popular and widely applied method is rule-based controls (RBC) based on conditional logic and heuristic rules developed by controls engineers and building operators. In rule based controls, the controls rely on predetermined set points, and a local control loop, such as a proportional-integrative-derivative (PID) loop, is used to maintain these set points. Heuristic controls are readily available, with ASHRAE Guideline 36 providing valuable recommendations on their use. However, these strategies, based on preset rules and heuristics, may not deliver near-optimal performance due to their lack of adaptability to specific building types, geographical lo-

cations, and fluctuating weather conditions. These controllers are moreover deficient in accounting for predicted data, such as weather forecasts, price signals, demand limiting signals and occupancy schedules. As a result, traditional controllers often fall short of optimal performance [6]. Furthermore, developing an efficient rule-based controller demands significant engineering time and expertise for calibration and performance tracking, especially to achieve objectives such as energy conservation and enhanced thermal comfort. Conventional building controls thus have limited use in performing multi-objective optimization and are unable to handle the growing needs of handling these added complexities.

Alternative to RBC, advanced control strategies, including model predictive control (MPC) can perform multi-objective optimization by considering future information, but they require extensive engineering effort for model development to accurately represent the thermal dynamics of the building energy systems [7]. Additionally, commercial buildings are by nature unique, and thus, developing custom models for each and every building would be very labor-intensive and therefore not feasible. Thus, MPC, despite its potential, has yet to be widely adopted in the building industry [8]. Thus we shift the focus on RL approaches as they can learn in a model-free fashion.

This paper is structured to ensure a thorough understanding of the subject. Initially, it presents a brief overview of reinforcement learning (RL), setting the stage for in-depth discussions. Following this, the study examines the machine learning (ML) models that are crucial to the development of surrogate models. These surrogate models are essential in enhancing the RL training process. The research further explores existing literature on the application of RL in building control, specifically focusing on the aspects of training duration and effort, making the reader appreciate the need to address the learning capabilities of RL to improve its potential of becoming practical in a real-world setting. It then proceeds to investigate various methodologies adopted in this study, leading to an analysis of the findings.

2 Reinforcement Learning

Reinforcement learning is a form of goal-driven learning to derive sequential decisions from its interaction with an environment. It can learn model-free and aims to maximize the long-term cumulative rewards it attains from the environment [9]. RL can be formulated as a Markov decision process (MDP), which makes it possible to apply a variety of algorithms to solve RL problems to find an optimal policy π in an environment. The MDP in RL can be formulated by the tuple (S, A, S', r, γ, P) where S is the current state of the environment, A is the control action, S' is the next state, r is the reward from the environment, γ is a factor within [0,1] that determines the trade-off between long term and short term goals of the RL controller and P is the transitional probability of the environment. RL has shown remarkable success in performing complex tasks like playing Atari 2600 video games [10] and the game of Go [11] at a human expert level, found success in fields like autonomous vehicles [12–14], robotic applications [15,16] and natural language processing (NLP) like ChatGPT [17].

In contrast to MPC, RL offers a compelling alternative by learning optimal actions through repeated interactions and continually refining its control policy. RL's adaptability to changing dynamics is a significant advantage. Despite of these benefits, RL faces challenges such as lengthy training periods and initial instability. The training duration can vary widely, from a few weeks [18] to up to 40 years [19], depending on the complexity of the building control problem. In practical scenarios, such extended training times and the associated discomfort and energy inefficiencies during early learning phases are untenable for control engineers and building occupants.

The aim of this research is to tackle the challenges of lengthy learning periods and early-stage instability that hinder RL's practical implementation and broad adoption. This study focuses on developing strategies to enhance sample efficiency and accelerate the learning process of an RL agent in real-time applications. A novel approach proposed here involves supplementing RL training with surrogate models derived from supervised ML techniques. These models assist the RL agent in its learning journey, reducing the likelihood of making evidently undesirable choices in the early stages of training. Examples of such undesirable actions might include increasing power consumption during a period of demand limitation or excessive heating/cooling during times when the building is unoccupied. This research seeks to streamline the learning process, making RL a more viable option for efficient and effective building control.

The next section provides a general introduction to ML along with the relevant ML techniques used for the surrogate models. The details of the use of the surrogate models and the approaches are mentioned later in Section 4.1.

3 Machine Learning Models

3.0.1 Random Forest. Random forests [20] are an ensemble learning method that combines multiple decision trees [21] to create a more accurate and robust prediction model. Decision trees are simple models that split the data into subsets based on the value of a single feature at each node and then predict the outcome based on the majority class in each subset. While decision trees are easy to interpret and implement, they tend to overfit the training data, which leads to poor generalization performance on unseen data.

Random forests overcome the limitations of decision trees by generating a large number of trees and introducing randomness in two ways: by selecting random subsets of the data for each tree and by randomly selecting a subset of the features at each node. This randomness creates diversity among the trees, which reduces the correlation among their predictions and improves their collective performance. We use a random forest model to predict the temperature and whole building power response from the building data.

3.0.2 Artificial Neural Network. An artificial neural network is a function whose mathematical model is inspired by the neurons of the brain. An artificial neural network (ANN) consists of the connections of an input layer, a hidden layer or layers, and an output layer. Each layer consists of several units called neurons. A node in an ANN receives inputs from other nodes, processes the input, and produces an output through an activation function. The nodes are interconnected by weights and biases, which can be adjusted by the training process to determine the strength of the signal of individual nodes. The combination of the signal outputs of all the nodes results in the overall behavior of the ANN. The training process involves adjusting the parameters (weights and biases of nodes) of the ANN to reduce a loss function between the prediction outputs from the network and the observations coming from actual data. The parameters are usually updated with a gradient descent algorithm. There are several types of ANN, but in this study, we are only concerned with deep neural networks (DNN).

Deep neural networks (DNN) [22] is a form of ANN where there are multiple hidden layers where each layer processes the output of the previous layer. They have proven to be highly effective in many complex problems. The multiple layers allow for the representation of more complex, abstract relationships in the data, enabling to modeling of non-linear relationships effectively between the input and the output variables. They can automatically learn useful features from raw data without requiring much pre-processing and feature engineering from large high-dimensional data spaces. The availability of computational resources has influenced and popularized the use of DNNs as they showcased their success in various implementations. Although they are successful in regression applications, the challenge lies in the interpretability of the model and overfitting. We use the DNN to predict the reward function r from the building data.

4 Review of Past Work

The purpose of the literature review is to highlight the issue of the training time needed for the RL agent to learn a near-optimal solution. There are mainly three types of literature available with RL applications on building controls: i) online learning: where the agent learns the optimal strategy while the system is controlled actively; ii) pre-training: an offline pre-training setup with a surrogate model (physics-based or data-driven), before it is deployed in a real building or environment and finally iii) transfer learning: this is where a trained agent is been applied to a similar environment. The drawback with online learning is that it is not possible to implement it in a real-world application due to exploratory and unstable behavior as well as long training times. The challenge with offline learning is that it still requires a considerable amount of data to build an accurate representation of the building environment. In transfer learning, an RL agent needs to be trained before it can be deployed to a similar environment and most of the literature takes training from one of the mentioned approaches (online or offline) and analyzes the adaptability of the agent in the new environment. Thus, the practicality of using transfer learning still depends on addressing the training issues faced by online and offline approaches in buildings and thus we have mainly shown literature on RL applications on building controls with offline and online approaches. The purpose of the literature is mainly to showcase the training time and method required by the RL agent to learn a near-optical policy in a building environment. We have highlighted the training times in bold in the paragraphs below to showcase the issue of the long training times faced.

Henze et al. [23] were early adaptors in applying model-free reinforcement learning (RL) to building energy systems utilizing a tabular Q-learning [24] algorithm to manage a thermal energy storage system. The RL agent needed **200 weeks and 900 weeks of training data** to match the performance of traditional control strategies under time-of-use pricing and real-time pricing schemes, respectively but couldn't surpass model predictive control methods. In further research, Liu et al. [25,26] with **6000** days of emulator training for managing temperature setpoints and energy storage was unable to outperform conventional strategies, facing challenges like emulator-reality mismatches and slow feedback from real data and increase in complexity with discretization of state and action space.Dalamagkidis et al. [27] using a temporal-difference (TD) RL algorithm [28] required **four years of training data** to match the performance of a conventional rule-based controller to maintain thermal comfort.

Advancements in computational resources, deep learning, and reinforcement learning (RL) algorithms have enabled research on applying deep RL to building energy systems. Yang et al.[29] with a batch deep Q-learning [30] agent took **three years** of data to surpass the rule-based controller. Li and Xia in [31] was able to get a good policy **26 weeks** of training data. Costanzo et al. [32] used a model-assisted training with extreme learning machines (ELM), achieving a good policy with **20 days of interaction**. Wei et al. [33] applied a deep RLC with **100 months of training data of summer days** on an EnergyPlus model, outperforming the rule-based policy with a energy savings range from 5% to 70%. Li et al. [34] utilized a deep-deterministic-policy-gradient (DDPG) [35] achieving 11% energy savings trained on **one year of simulation data** generated from random control policy from EnergyPlus simulation. [36] using a Deep Q-Network (DQN) agent required **five years of data** inclusive of one month's data of random control actions, and two months' data of rule-based policy, with the rest of the days trained in an on-policy fashion to achieve a good policy.

The following sections reports literature that uses some hybrid learning methods where a surrogate model is used to assist with the RL training. Chen et. al [37] adapted a differential MPC method from [38]. A policy is generated from the MPC problem of a linear model whose system dynamics are determined simultaneously from the **historical rule-based data of 20 days**. This policy is then imitated by the RL agent and fine-tuned with online training. Although it effectively reduced the online training time and improved performance over the existing policy before online interaction, the challenge lies in determining an accurate linear model of a real building with a very limited data set of rule-based data that does not explore the state space.

Zou et al. [39] utilized LSTM models trained on **one year of building data** to improve deep RLC training with a DDPG agent. They achieved a 30% reduction in energy consumption while maintaining a 10% dissatisfaction rate. Costanzo et al. [40] employed model-assisted batch reinforcement learning (MABRL) with fitted Q iteration, utilizing multiple instances of single-layer, single-output extreme learning machine (ELM) models to predict temperature changes. The agent achieved a decent policy in 10 days and approached 90% of mathematical optimality within 20 days. Arroyo et al. [41] integrated RL with MPC, leveraging the RL agent's value function in the MPC controller's objective. Despite improved performance, accurate model development remains crucial and challenging, particularly with limited data from rule-based strategies. They utilized **4 weeks of data** for model development, facing difficulties due to insufficient excitation in rule-based building operational data.

The majority of the research on RL applications on building controls focuses on the final performance of the RL agent with an online or offline approach. There is a lack of research on addressing challenges of long, unstable initial training time and the model required to train the agent. Few papers including [37,40–42] try to address the learning speed of the RL agent, while [37,41] uses a combined RL-MPC approach which brings in both the positives along with the negatives of both worlds of MPC and RL. The research extends on the concepts presented in [40,42], comparing several model-assisted data-driven online learning approaches addressing the learning challenges as mentioned before.

4.1 Approach and its Novelty.

This work presents an alternative data-driven method to address the practical issues faced with RL applications. The advantage of RL over other control approaches is its ability to learn purely from environmental data from its control actions. This is particularly appealing in building energy management systems as building systems are very varied, and thus, customized accurate models are not feasible to make for every building to set up an optimization problem. RLC, despite of its model-free advantages, suffers from long training times, sample inefficiency, and unstable early exploratory behavior which makes it difficult for a plug-and-play approach. Most of the existing literature on RL on building applications uses simplified models with simple objectives, showcasing the performance gain at the end of training on the same simulated model. However, these simplified models are not representative of real-world settings, nor can they afford the long training times with full exploration.

The model development requires months of engineering effort, time, and money to accurately model a building system using various system identification techniques, most commonly being a linear time-invariant (LTI) metamodel. But over time, the building system dynamics may change, requiring redevelopment of the building model. Here, we envision an approach where the RL interacts and learns directly from the building environment in an online fashion, addressing the common problems of the pure online and offline mentioned above, assisted with data-driven surrogate models. The issue of long training wait times and unstable behavior in online training, as well as the development of accurate models for offline training, can be avoided with this approach. The surrogate models are built off the data gathered from the RL interaction. Although these models are inaccurate in the beginning, as they are built on a very limited dataset, they assist the RL in guiding them towards better decisions and preventing unnecessary and repeated high-penalty actions, when compared to a pure online approach.

We use state-of-the-art deep machine learning techniques in place of traditional system identification methods for our surrogate model development. Deep learning techniques serve as a versatile

approach for mapping nonlinear complex relationships from inputs to outputs without knowing the underlying system dynamics. Deep learning can handle large amounts of data without needing manual feature selection processes, making it suitable for a wider range of applications in relation to traditional system identification processes at the cost of requiring high computation [43]. We utilize the deep-learning model to predict the rewards and a random forest model for the building temperature response and the whole building power consumption. The surrogate models are used to analyze the consequences of taking some sequence of actions in the vicinity of the rule-based actions in the future and use that information to assist the agent in converging to a near-optimal policy faster than the traditional approach.

5 Reinforcement Learning Algorithm Used

In reinforcement learning (RL), a policy refers to a strategy or a set of rules that an agent uses to make decisions in an environment and is essentially a mapping between states and actions. One of the main challenges in reinforcement learning is the exploration-exploitation trade-off: the agent must explore enough to discover a good policy, but also exploit that policy once it has been discovered to maximize reward. Here, we use the soft actor-critic (SAC) algorithm as the controller agent because it has shown success in many continuous control tasks and exhibits a good exploration-exploration tradeoff. One way that SAC addresses this challenge is by incorporating an entropy term in its value function SAC also incorporates a soft-Q policy which modifies the traditional Q-function to include the entropy term, thus aiming to maximize a trade-off between expected return and entropy. This encourages the agent to explore a wider range of actions and can help prevent premature convergence to suboptimal policies.

The core objective of SAC is to maximize the expected sum of rewards augmented by entropy. The objective function for a policy π is given below in Equation 1 by:

$$J(\pi) = \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(r(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))\right)\right] \quad (1)$$

where:

- $r(s_t, a_t)$ is the reward received after taking action a_t in state s_t.
- γ is the discount factor.
- α is the temperature parameter that scales the importance of the entropy term, effectively controlling the trade-off between exploration and exploitation.
- $\mathcal{H}(\pi(\cdot|s_t))$ is the entropy of policy π at state s_t, promoting exploration by penalizing low-entropy policies, and is mathematically quantified as $-\mathbb{E}_{a\sim\pi(\cdot|s)}[\log\pi(a|s)]$.
- ρ_π denotes the state-action distribution under policy π.

SAC employs two Q-networks, $Q_{\phi_1}(s,a)$ and $Q_{\phi_2}(s,a)$, with parameters ϕ_1 and ϕ_2, respectively. These networks estimate the return (sum of future rewards) from taking an action a in state s and following the policy thereafter. Using two Q-networks helps in reducing the overestimation bias of the Q-values.The Q-functions are updated to minimize the mean squared Bellman error (MSBE). The target $y(r_t, s_{t+1})$ for the Q-function update is given by $r_t + \gamma\left(\min_{i=1,2} Q_{\phi_i'}(s_{t+1}, a_{t+1}) - \alpha\log\pi_\theta(a_{t+1}|s_{t+1})\right)$, where $Q_{\phi_i'}$ represents the target Q-networks which are periodically updated with the weights of the Q-networks to stabilize training. The policy is updated by minimizing the following objective:

$$J(\pi) = \mathbb{E}_{s_t\sim\rho_\pi, a_t\sim\pi}\left[\alpha\log(\pi_\theta(a_t|s_t)) - Q_\phi(s_t,a_t)\right] \quad (2)$$

Sample inefficiency is a common concern in RL where it refers to a situation in which a learning algorithm requires a large number of training examples or interactions with the environment to achieve satisfactory performance or policy convergence. One of the key advantages of SAC is its sample efficiency. Compared to other popular reinforcement learning algorithms, such as DQN and actor-critic (A3C), SAC is able to learn policies for continuous control tasks using far fewer samples. This is particularly important in real-world applications, where the cost of collecting samples can be high. We refer the reader to [44] for in-depth details about SAC.

6 Building Framework Used

In this research, an open-source building performance simulation test bed, the Advanced Controls Test Bed (ACTB) [45] is utilized. This allows the user to utilize a high-fidelity building Spawn of EnergyPlus (SoEP) model, which combines an envelope model in EnergyPlus with the equation-based HVAC components written in the Modelica language. The RL agent utilizes the OpenAI Gym interface of the ACTB for the RLC training and evaluation. Both supervisory set-point and low-level controls can be implemented in the ACTB with a controller written in any language. The controls of the HVAC systems in the ACTB implement ASHRAE Guideline 36 when not overridden with an external controller script. A solar PV and battery model are also incorporated, the details of which are described below.

6.1 Implementation. This work demonstrates the use of various approaches of harnessing surrogate models to steer the initial exploration process. We have implemented two surrogate models in this guided exploration approach: 1) Building-centric surrogate model (M_B) and 2) Reward-centric surrogate models (M_R). The building-centric model (M_B) predicts the temperature response of the detailed SoEP building model as well as the whole building power. The building-centric models help to determine the subsequent state, which is necessary to compute the rewards via the reward-centric model M_R. Figure 1 explains the approach. These surrogate models are constantly updated with fresh data generated from online RL interactions. We keep the training episodes of all the approaches to 60 episodes (approximately two months of training time), where each episode is a day. We detail the diverse strategies that we investigated below:

6.1.1 Approach 1: Baseline Case with Pure RL. This approach presents the baseline scenario of utilizing pure reinforcement learning training without the help of any surrogate models. The model starts with no prior knowledge nor is it being guided by the surrogate models in its early stages. The learning process is reliant fully on an exploration of the environment in real-time. This scenario serves as the benchmark case against which other approaches are compared, particularly in terms of unstable exploratory behavior during a fixed training time and the end performance at the end of this training period. We will refer to this approach as $Ag^{Dir}_{RL,1}$.

6.1.2 Approach 2: Short-Term Guidance with Surrogate Model Exploration. This approach begins with a guided strategy, wherein surrogate models are employed to generate artificial paths and ascertain which actions yield the most favorable outcomes. We produce n artificial roll-outs of t time-steps from the current state and time. We select the first action a_{guid}, which leads to the least penalty-objective of $\sum_t r(s,a)^t$. The executed action is $a_{exe} = \beta_1 a_{guid} + \beta_2 a_{RL}$, which is a weighted action between the guided action from the surrogate models and the action from the RL agent. The weighted action helps to control the reliance of the guided action. During the artificial path generation from the current state, we use a modified action for artificial exploration given by $\beta_1 a^{exp} + \beta_2 a^{RL}$, where $0 \le \beta_1, \beta_2 \le 1$, $\beta_1 + \beta_2 = 1$ and a^{exp} is a random action. We keep β_1 at a value of 0.95 at the start and reduce it gradually to 0 in 30 episodes of training with $n = 50$ samples.

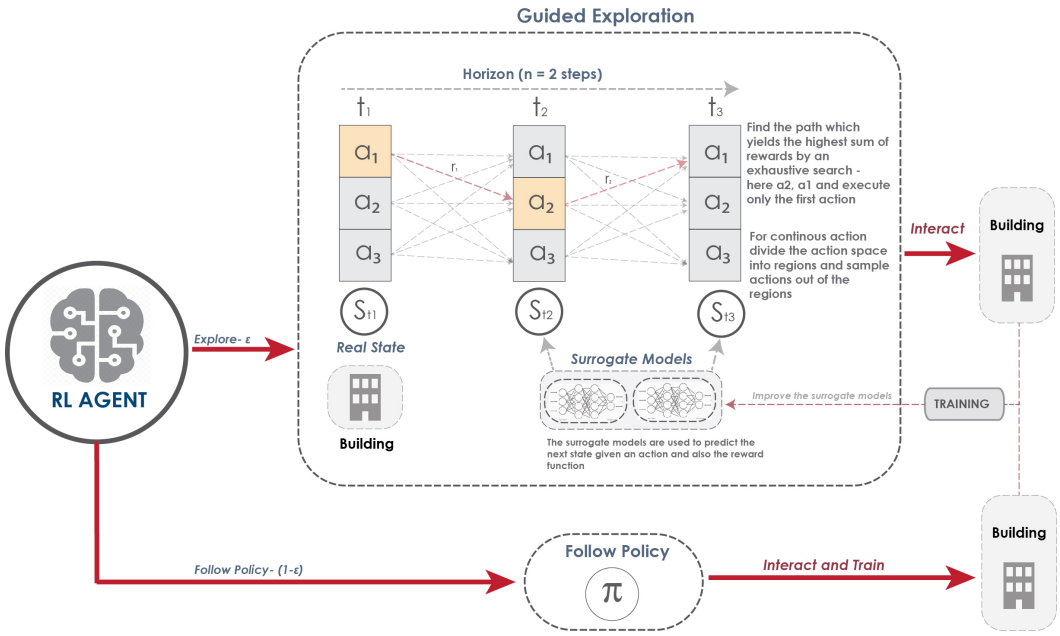

Fig. 1 Online Learning with guided exploration

A high initial value of β_1 is chosen to encourage more exploration of the state space in an artificial manner, especially when the RL agent is at its nascent stage of learning. A low value of β_1 in the later stages keeps the exploration near the RL agent's policy and artificially explores the environment in a constrained manner to find a better policy. In Approach 2, the short-term guidance uses only a roll-out of one-time step with $t = 1$. We continue normal direct training after 30 episodes with no guidance from the surrogate models. This approach will be referred to as $Ag^{Guid}_{RL,2}$.

6.1.3 Approach 3: Short-Term and Long-Term Guidance with Surrogate Model Exploration. This approach is similar to the previous approach for the first episodes of training but after 20 episodes, we generate artificial samples with roll-outs of $t = 2$ from the current state from episodes 20 to 30; $t = 3$ from episodes 30 to 60. Here, we also train the agent on the artificial trajectories generated from the surrogate models. This approach will be referred as $Ag^{Guid}_{RL,3}$.

6.1.4 Approach 4: Short-Term and Long-Term Guidance with Surrogate Model Exploration around Rule-Based Actions. The fourth approach uses rule-based control strategies to curtail exploration where the agent's actions are overridden with rule-based control actions with a probability of 70% from episode 1, reduced gradually to 2% in and after episode 20. Here, artificial trajectories are generated, and the RL agent is trained on these artificial trajectories at every step. The artificial trajectories are generated around a rule-based policy such that $a_{art} = a_{RBC} + \sigma$ where a_{RBC} is the rule-based action, σ is a noise parameter sampled from a uniform distribution $U(-0.1, 0.1)$. We commence with roll-outs of $t = 1$ from the current state from episodes 1 to 20; $t = 3$ from episodes 20 to 30; $t = 6$ from episodes 30 to 40, and $t = 12$ from episodes 40 to 60. This approach will be referred to as $Ag^{Guid}_{RL,4}$. We increase the time frame in the later episodes as the surrogate models are inaccurate in the early stages.

6.1.5 Approach 5: Imitation Learning. This approach employs imitation learning, as detailed in [46]. In this case, the actor-network is trained in a supervisory fashion to imitate the actions of the supervisory rule-based actions. Artificial data is generated in the tuple form of (S_{RL}, a) where a is the corresponding rule-based action for the state S_{RL}. The data is generated so that it covers the state-space uniformly with sufficient data points. Then, we perform a supervised learning with the policy network of the SAC RL agent to mimic the rule-based policy given the state S_{RL}. This helps the RLC agent start by following the rule-based policy and prevent potentially harmful exploratory actions in the early training phases. We generated data equivalent to 600 days of data for the imitation learning training. The supervisory training is stopped until it achieves an accuracy of 95% on the validation data, where the generated policy is within ± 0.2 °C. This is essentially a supervisory training, where the actor NN is trained to map certain states to actions which are the rule-base controls set-point.

An artificial dataset is created following rule-based actions to train the policy network of the RL agent. The policy network of the agent is trained on this artificial dataset to mimic the rule-based actions. The artificial dataset essentially consists of tuples of states and their corresponding rule-based action. Then, supervisory training is conducted to map the states to the rule-based actions. The artificial dataset generated consisted of two years of data. The training of the policy network is kept on hold for the first five days of interaction, and only the value networks are trained. This approach will be referred to as $Ag^{Imit}_{RL,5}$.

6.1.6 Approach 6: Combined Short-Term Guidance and Imitation Learning approach. In this approach the RL agent is taught to imitate the rule-based actions beforehand as it is with the previous case but is assisted with the surrogate model. This approach is very similar to the approach $Ag^{Guid}_{RL,2}$ (6.1.2), where the artificial trajectories are generated with $\beta_1 a^{exp} + \beta_2 a^{RL}$, but here action a^{RL} from the RL agent starts with the imitated rule-based policy, and thus the trajectories are developed around the rule-based policy. The difference between this approach with Approach 4 is that although the exploration of the artificial trajectories is around the rule-based controls, here the agent is primed itself to follow the imitated policy whereas in $Ag^{Guid}_{RL,2}$, the agent is trained on the artificial trajectories around the rule-based policy. This approach will be referred to as $Ag^{Imit+Guid}_{RL,6}$.

7 Experimental Design

7.1 Environmental Setup. Supervisory set-point control is a common control approach for building systems world where its sequence of operation can affect the indoor environmental conditions, energy consumption, and electric demand of the building.

The building model used here is Spawn model of the U.S. Department of Energy's (DOE) Reference Small Office Building [47], which is composed of four perimeter zones and one core zone in a single story with a floor-to-ceiling height of 3 m. The floor area of the building is 511 m^2 and the building is located in Chicago. The HVAC systems of this building consist of a constant air volume AHU composed of a gas-fired heating coil, a single-stage direct expansion cooling coil, an outside air damper without an economizer, and a constant volume fan. The five thermal zones are each supplied by a packaged rooftop unit (RTU).

The southern roof and the northern roof have solar PV cells installed. The pitch angle of the southern and the northern roof is 46% with an area of 150 m^2, while the area of the east and west roofs has an area of 101.9 m^2 with a pitch angle of 18.6°. We assume that 95% of south, east, and west are covered with solar PV cells, which amounts to 143 m^2 of solar PV panels on the north roof and 97 m^2 each on the east and west roof. Each module has an area of 1.7 m^2 leading to a total of 84 modules in the PV panel, each on the south wall and north wall, and 59 modules each on the east and west PV panel. There are also 300 modules of PV installed on car port that contribute to the PV generation.

The demand response (DR) events occur randomly between 2:30 PM and 4:00 PM and can last for a duration between 1.5 and 3.5 hours. The DR event type is a demand limiting program, where the demand limit is set to be 20 kW. The peak whole building power is found to be 80 kW during the afternoon hours. The demand limit is determined from the peak.

The specification of the batteries is based on the Tesla PowerWall with an energy capacity of 13.5 kWh; we utilize six Tesla PowerWall batteries which make up a total storage capacity of 81 kWh.

7.2 States Considered for RL agent. The states S_{RL} considered for the feedback of the RL agent are:

- T_{zi}: Mean zone temperature of all the zones in the building.
- t_h: Current time in hours.
- T_{OA_0}: Current outside air temperature.
- T_{OA_1}: Forecasted outside air temperature in 1 hour.
- H_{Glo_0}: Current horizontal global irradiation.
- H_{Glo_1}: Forecasted horizontal global irradiation in 1 hour.
- DR_{cnt}: A countdown time signal from the start of the DR event which is notified 4 hours ahead of the DR start time, and returns zero during other times of the day.
- DR_{curr}: A binary signal indicating if the current time falls in the DR event for the day. This returns 1 during a DR event, returns the duration of the expected DR event 4 hours before the start of the DR time, and zero during the other times of the day.
- B_{SOC}: Current state-of-charge of the battery.

7.3 Actions Considered for RL Agent. The action vector A_{RL} of the RL agent is $[T_{zi}^{sp}, b_a]$, which is described below:

- **Supervisory Setpoint** (T_{zi}^{sp}): This is the supervisory temperature set-point action passed to all the zones in the buildings. The agent outputs a value between -1 and 1, which is scaled to a supervisory set-point temperature between 15°C and 30°C.
- **Battery Action** (b_a): The battery control action b_a operates within the range of $[-1, 1]$. This control action determines either the charging demand (when $b_a > 0$) or the discharging demand (when $b_a \geq 0$). The charging demand from a single battery is given by $b_a B_{max}^{cha}$ while the discharging demand is given by $b_a B_{max}^{disc}$, where B_{max}^{cha} and B_{max}^{disc} are the maximum charging and discharging capacity of a single battery respectively.The charging and the discharging modes of the battery are explained below:

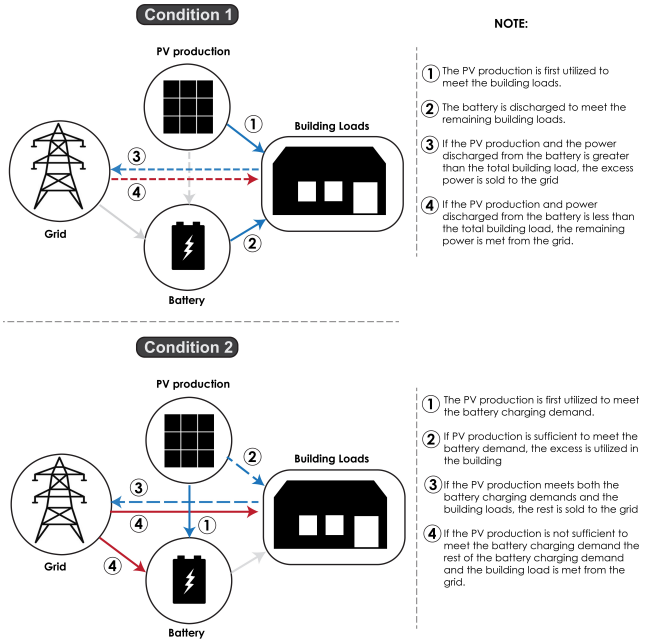

Fig. 2 Diagram showing different control modes of the battery with grid, solar PV, and building

The following highlights the working of the two modes of the battery - the discharging and the charging state.

- **Discharging Condition** [$b_a \in$ [-1,0)]: In this condition, the PV production is first utilized to meet the building loads. The discharging power is used to meet the remaining building loads. If PV production and the power discharged from the battery meet the total building load, the remaining power is sold to the grid. In the case that the PV production and the discharged power from the battery fail to meet the building load, the remaining power is taken from the grid.
- **Charging Condition** [$b_a \in$ [0,1]]: In this condition, the battery first charges from the PV production and the grid. First, the PV production is utilized to meet the battery charging demand, and if this PV production is greater than the battery charging demand, the rest is utilized to meet the building loads. If the PV production is less than the battery charging demand the remaining battery demand and the building load are met from the grid. Any available PV production during this time step is first used to meet the building loads. If the available PV production is greater than the building load, the rest is curtailed. The battery cannot charge and discharge at the same time. These different modes of operation, as mentioned above are explained in the Figure. 2.

7.4 Rule-based Control Strategies (Baseline Control).

- The supervisory set-point action has bounds between 15°C and 30°C and can take any setpoint within this bound at the control time-step of 5 minutes.
- The supervisory temperature control strategy follows the lower thermal comfort bound as mentioned in Section 7.5, except for the conditions mentioned below.

- The supervisory temperature control strategy follows the upper thermal comfort bound one hour before the start of the DR event to preheat the building before the DR event.
- The battery (b_a) is charged at its full capacity rate two hours before the DR event.
- The battery (b_a) is discharged at its full capacity during the DR event.
- During the DR event if the charge of the battery is depleted, the temperature supervisory setpoint is reduced by 1°C.

7.5 Reward Formulation. The reward formulation is expressed in Equation 3. Each zonal controller receives rewards coming from the thermal discomfort and energy consumption of its zone controlled and the power penalty of the whole building. The reward penalties are monetized to a price-based objective. The price for thermal discomfort is based on the assumption of a desk job salary in Chicago and its relation to the reduced productivity of the employees. This means the penalty for thermal comfort in the building is proportional to the occupancy of the zones. The thermal comfort bound is between 21°C and 24°C during the occupied hours of 6:00 a.m. to 10:00 p.m and 15°C and 30°C during the remaining unoccupied hours of the day.

Since the building is an office building with white-collar jobs, the average salary of the employees was considered to be slightly higher than the average Chicago salary of $74,000/year [48]. Here, the salary was assumed to be $80,000/year which roughly rounded up to $40/hr. From [49], it was assumed for a 1°C temperature rise, the productivity decreases by 2%. This resulted in a thermal discomfort cost of $0.8/Kh. The average cost for commercial electricity is taken to be $0.0405/kWh [50] and the demand charge was taken to be $7.89 per kW of demand limit violation observed for each five-minute interval.

When the key performance indicators (KPI) are converted to price-based penalties, the thermal discomfort price is much higher when compared to the energy price. Thus, the w_i weights are included in the penalty objective to scale the price-based penalties into values, such that each KPI have similar importance in the objective function. The weights w_i are negative as these are penalty costs that the agent aims to minimize.

$$r = p_{disc} T_{disc} occ + p_e E_{Net} + p_{dr}(P - P_t) \tag{3}$$

where:

$$\begin{aligned} T_{disc} &= \text{Thermal discomfort penalty per time step [Kh]} \\ E_{Net} &= \text{Net energy consumption per time step [kWh]} \\ P &= \text{Average power per step [kW]} \\ P_t &= \text{Power threshold during DR Event [kW]} \\ p_{disc} &= \text{Price for thermal discomfort [\$/Kh]} \\ p_e &= \text{Linear hyper-parameter for power penalty [\$/kWh]} \\ p_{dr} &= \text{Demand charge [\$/kW]} \\ occ &= \text{Occupancy [1]} \end{aligned} \tag{4}$$

Zone i controller receives the reward based on the thermal discomfort and energy consumption of Zone i, but shares the power penalty term of the whole building. The power threshold P_t during the DR event is taken to be 20 kW.

7.6 Surrogate Models.

7.6.1 Building-Centric Surrogate Model States (S_B). The input states considered for the surrogate model M_B are:

- T_{zi}^{t-1}: Mean zone temperature of all the zones in the building, one-time step $(t-1)$ back.
- T_{zi}^{t}: Mean zone temperature of all the zones in the building at the current time step t.
- t_h: Current time in hours.
- T_{OA}^{t}: Outside air temperature at current time step t.
- H_{Glo}^{t}: Forecasted outside air temperature in next time step.
- a_{sp}: The supervisory set-point temperature action.

The output prediction of the surrogate model are:

- ΔT: Change in mean zone temperature during the current time step
- P: Whole building power demand

7.6.2 Reward-Centric Surrogate Model States. The reward-centric surrogate model has a DNN implementation with 9 input states and 1 output state to predict the reward function. The input states considered for the surrogate model M_R are:

- T_{zi}^{t}: Mean zone temperature of all the zones in the building at current time step t.
- t_h: Current time in hours.
- T_{OA}^{t}: Outside air temperature at current time step t.
- H_{Glo}^{t}: Forecasted solar radiation at current time step t.
- DR_{cnt}: Countdown time signal in hours from the start of the DR event as mentioned in Section 7.3.
- DR_{curr}: Binary indicator if the current time lies in a DR event as mentioned in Section 7.3.
- T_{zi}^{sp}: Supervisory temperature set-point control action as mentioned in Section 7.3.
- b_a: Control action of the battery mentioned in Section 7.3.

The output prediction of the surrogate model is:

- r: The rewards for landing in the current state due to the control action taken in the previous step.

We name the state for the building-centric surrogate model S_R such as $S_R = [T_{zi}^{t}, t_h, T_{OA}^{t}, H_{Glo}^{t}, DR_{cnt}, DR_{curr}, T_{zi}^{sp}, b_a]$. All the states were normalized into a continuous value between 0 and 1 except for the control states T_{sp}, where the normalization is done between -1 and 1.

Table 1 Neural network structure

Layers	Shape	Activation
M_R		
input	(9,-)	*tanh*
hidden 1	(1200,-)	*tanh*
hidden 2	(1800,-)	*relu*
hidden 3	(2000,-)	*relu*
hidden 4	(1800,-)	*linear*
hidden 5	(1800,-)	*linear*
output	(1)	*linear*

8 Results

The training progress of the different approaches is shown in Figure 12. In Figure 12, the lines are representative of the 10-day mean cost incurred from episodes 1 to 60. The red-dotted line is the direct RL approach $Ag_{RL,1}^{Dir}$, which does not use any guidance from the surrogate models and serves as the baseline case. $Ag_{RL,1}^{Dir}$ without any guidance incurs high costs in the early episodes as it explores the environment. Approaches $Ag_{RL,4}^{Guid}$ and

$Ag^{Imit+Guid}_{RL,6}$ had better training progress as it incurs lower costs in the early stages and have more or less the same cost with the direct training approach $Ag^{Dir}_{RL,1}$ during the middle and end of the training period. Figure 12 shows that most of the approaches are successful in preventing high penalties in the early stages of training but may incur slightly higher costs than the baseline case during the middle of the training, as we see $Ag^{Imit}_{RL,5}$ incur a slightly higher cost from the period of 15 to 35 training episodes. However, none of the surrogate model approaches incur as high costs as the direct approach in the period from 1-15 episodes, which proves that the model-assisted methods have improved the training efficiency of RL compared to the baseline case $Ag^{Dir}_{RL,1}$ which does not use any guidance from the surrogate models.

There are two types of imitation learning approaches used in the research which are shown in the bottom plot of Figure 12. The pure imitation approach $Ag^{Imit}_{RL,5}$ although has lower initial costs performs poorly during the mid-training period. The combined imitation and guided approach $Ag^{Imit+Guid}_{RL,6}$ avoids this poor mid-training behavior. All the model-assisted approaches avoid taking disastrous action repeatedly once it has encountered the negative consequence of a particular action in a similar state s.

Figure 4 demonstrates the early performance of the RL agent $Ag^{Dir}_{RL,1}$ in the early stages of training with no guidance provided from the surrogate models. The top plot displays the mean temperature across five zones, with the minimum and maximum temperatures illustrated in light blue colors and the supervisory control set-point marked by red-dotted lines. The second plot highlights the whole building power, depicted in red, and the net-building power in green. The third plot portrays the power sold back to the grid as well as the available PV generation from installed solar panels. The fourth plot showcases the battery's charging and discharging demand along with the state of charge of the battery, expressed in kWh.

Figures 6 and 7 shows the performance of the agent at the beginning and the end of the agent $Ag^{Guid}_{RL,4}$. It can be seen, compared to the first-day performance of $Ag^{Dir}_{RL,1}$, that the agent's erratic behavior is significantly reduced as the agent's actions are overridden with a higher probability with the rule-based actions. This can also be seen with $Ag^{Guid}_{RL,2}$ in Figure 5, where the agent's performance in the first episode is reasonable. With feedback from the environment, limited exploration, and also training on artificial exploration around rule-based control actions, the agent learns some of the expected behavior of keeping the temperature between the comfort bounds, reducing power consumption during the DR event, charging the battery before the expected DR event and discharging it during the DR event within a span of 2 months of training interaction.

Figures 8 and 9 show the performance of the agent at the beginning and the end of the agent $Ag^{Guid}_{RL,6}$. Since instead of the actions being overridden with a rule-based policy and the RL agent starts with the rule-based imitated, it displays a better performance than the approach $Ag^{Dir}_{RL,4}$.

Figures 11 and 10 present the performance of agents $Ag^{Dir}_{RL,1}$ and $Ag^{Imit+Guid}_{RL,6}$ on test days, following the completion of the 60-day training period. The illustrations indicate that the controller has learned to maintain the temperature within comfortable boundaries. Furthermore, it has learned to charge the battery prior to demand response events and discharge it during these events. In instances where the battery is depleted during a DR event, the controller has learned to adjust the supervisory set-point downwards to reduce the heating power demand. In doing so, it accepts some thermal discomfort costs to avoid the higher costs associated with the excessive demand charges. From the figures, we can conclude that the agent still needs to learn to lower the setpoint during the unoccupied hours during the later part of the day from 10 PM to 12 midnight. As the agent has been trained for a fixed duration, it is essential for it to engage in additional interactions with the environment in order to acquire a closer-to-optimal policy, which is evident from all the figures at the end of the training performance.

The performance of the approaches on unseen test days is presented in Table 2, which highlights the costs incurred for each objective like energy consumed and sold from the grid, thermal discomfort, and demand response in dollar amount.

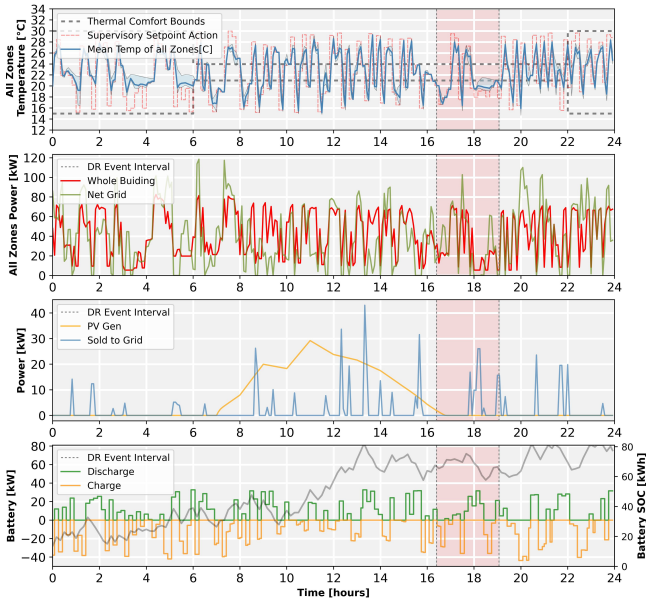

Fig. 3 RL agent $Ag^{Dir}_{RL,1}$ performance during the initial exploration stage on first training day

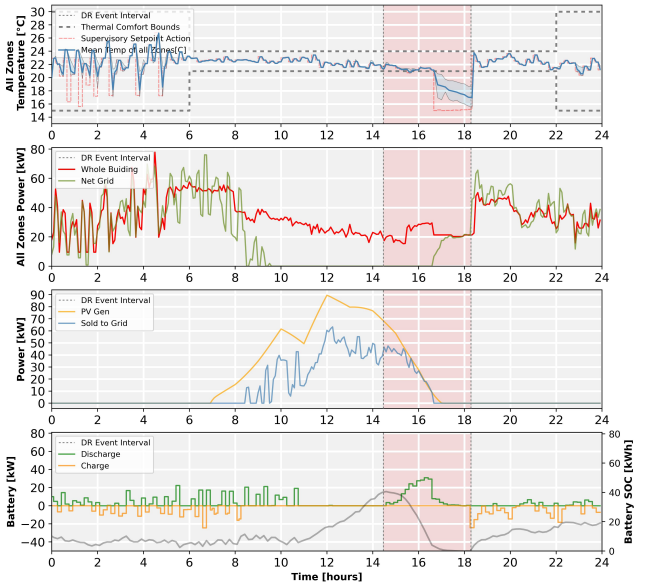

Fig. 4 RL agent $Ag^{Dir}_{RL,1}$ performance at end of training

9 Discussions and Conclusions

This work showcases a data-driven approach featuring an RLC agent in building energy applications, suggesting potential for real-world implementation in a building where an RL agent is directly

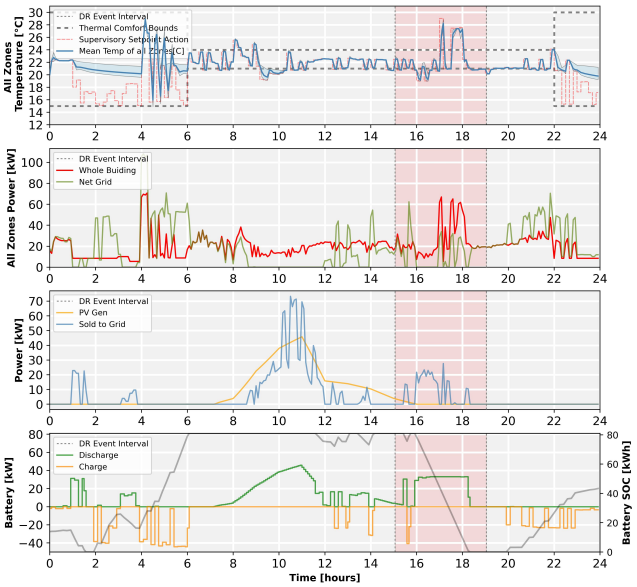

Fig. 5 RL agent $Ag^{Guid}_{RL,2}$ performance on first training day

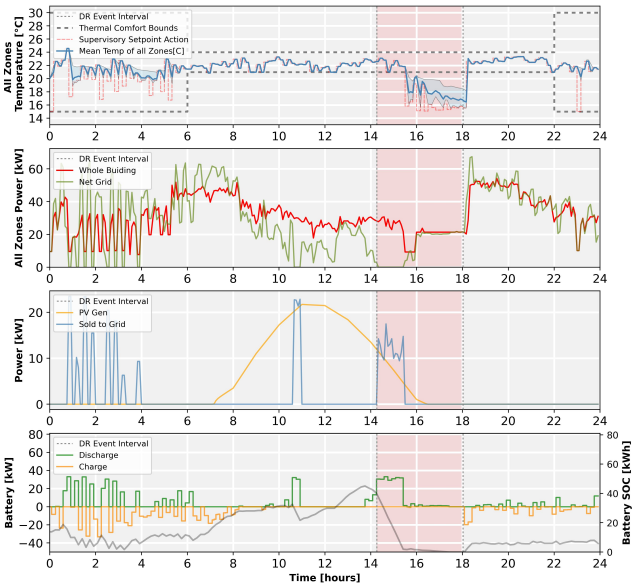

Fig. 7 RL agent $Ag^{Guid}_{RL,4}$ performance at end of training

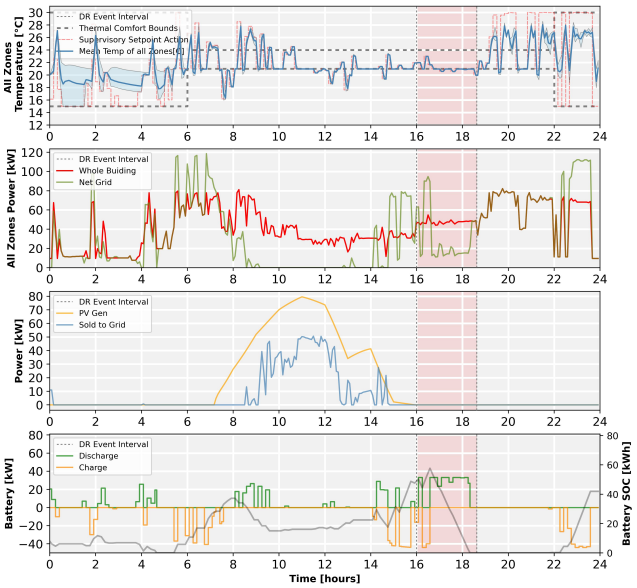

Fig. 6 RL agent $Ag^{Guid}_{RL,4}$ performance on first training day

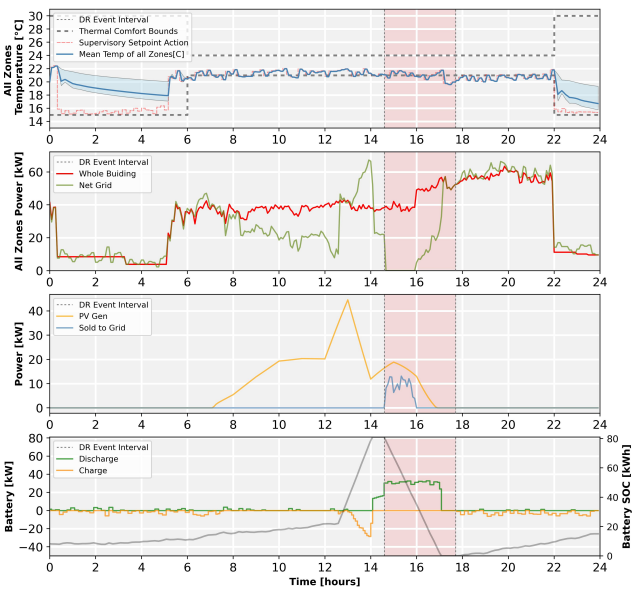

Fig. 8 RL agent $Ag^{Imit+Guid}_{RL,6}$ performance on first training day

controlling and learning in an online fashion. It sidesteps the conventional methods of linear dynamic system identification, which demand substantial effort in data collection, constructing, and validating models from limited data sets. Each building is unique, and thus, a solution developed for one building may not be applicable to other buildings, thus making optimization a challenging problem in relation to buildings.

This research leverages state-of-the-art machine learning regression techniques to learn the system dynamics and reward function in an online fashion, which is then utilized to reduce the training time and exploratory behavior of the RLC applications in buildings. Along with several machine learning techniques, we also use simple domain knowledge in the form of rule-based control strategies that the approaches utilized in the initial stages and then modify these strategies with further interaction from the real building. These strategies aid in reducing the early exploratory behavior of the agent as well as training times. The surrogate models assist in reducing the probability of taking actions that lead to higher costs. This is due to the fact that the surrogate models learn the reward functions more rapidly than the RL agent learns the value functions of the states of the optimal policy. The surrogate models are then utilized to generate some artificial trajectories and their subsequent returns from the environment for each trajectory. Once the agent has encountered a high penalty $r(s_p, a_p)$ by taking action a_p in a particular state s_p, the artificial exploration with the surrogate models will try to avoid a similar state-action pair, i.e., $(s, a) \sim (s_p, a_p)$. The models serve as an effective deterrent

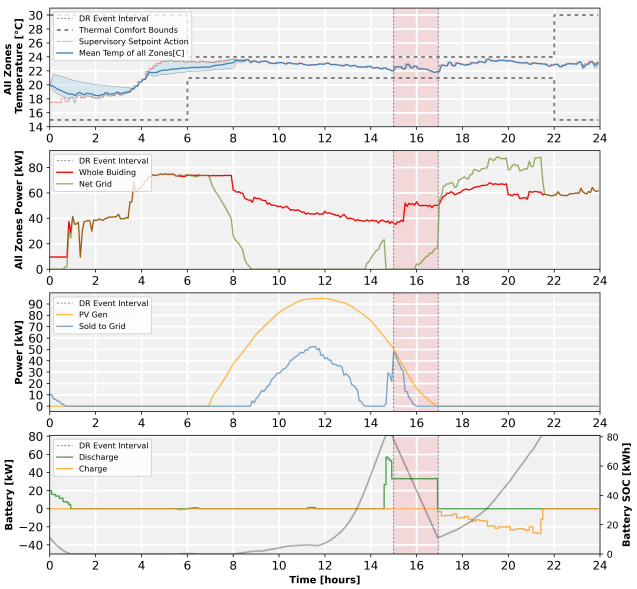

Fig. 9 RL agent $Ag_{RL,6}^{Imit+Guid}$ performance at end of training

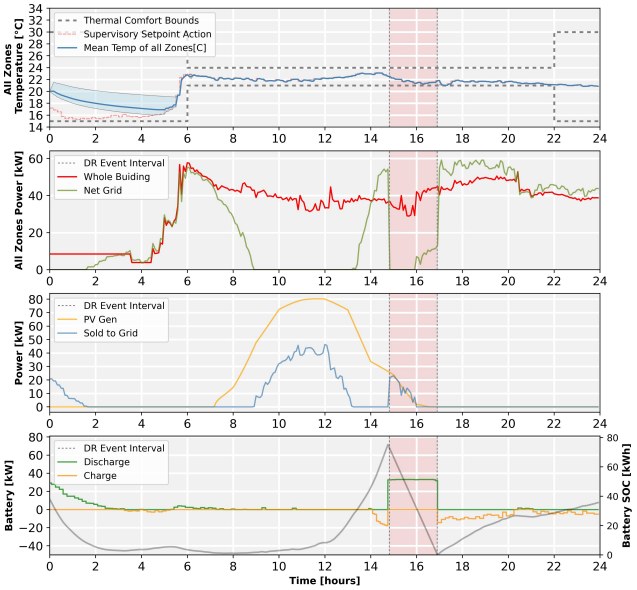

Fig. 10 Performance of the RL agent $Ag_{RL,6}^{Imit+Guid}$ on a test day, after end of training

against high-penalty actions when suggested by the RL agent and override it to an alternative better control strategy until the RL agent has a better estimate of the value functions.

We presented several systematic studies of different approaches aimed at reducing the probability of selecting exploratory actions for the early stages of learning for the RL controller. The combination of imitation learning and short-term guided exploration ($Ag_{RL,6}^{Imit+Guid}$) performed the best, as one can see in the training progress plot of Figure 12. The combination of imitation learning and the surrogate model approach helps to explore the environment artificially and nudge the policy towards a better control policy without making bad decisions repeatedly. This improves the training stability of the approach while significantly reducing high-

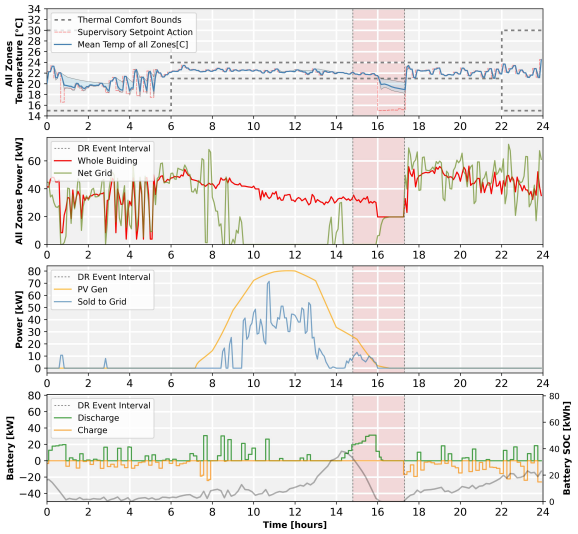

Fig. 11 Performance of RL agent $Ag_{RL,1}^{Dir}$ on a test day, after the training period

Table 2 Comparison of Results after training

Approach	E	E_{sold}	T_{disc}	DR	Tot[$]
Ag_{RBC}	58.2	26.2	93	1559.8	1652.7
$Ag_{RL,1}^{Dir}$	78.4	24.9	608.7	321.1	983.2
$Ag_{RL,2}^{Guid}$	78.8	28.3	683.7	179.1	913.2
$Ag_{RL,3}^{Guid}$	87.6	28.0	166.3	4369.6	4595.5
$Ag_{RL,4}^{Guid}$	86.0	28.4	763.3	278.6	1099.5
$Ag_{RL,5}^{Imit}$	71.8	23.2	4.4	1265.6	1318.6
$Ag_{RL,6}^{Imit+Guid}$	77.9	42.3	898.4	146.2	1080.1

[1] E: Energy consumption costs; E_{sold}: Energy sold.
[2] DR: Demand response costs
[3] T_{disc}: Thermal discomfort costs
[4] Ag: Controller agent
[5] Dir: Direct training approach with no guidance
[6] Off: Offline training approach
[7] $Imit$: Imitation learning approach
[8] $Imit+Guid$: Combined imitation learning and guided learning approach
[9] Tot: Total penalty cost

penalty actions during the early stage. However, the performance of this approach depends on the quality of the surrogate models used for the guidance.

The pure imitation learning approach $Ag_{RL,5}^{Imit}$ exhibited poor performance during the middle section of the training period as the agent was not guided by any surrogate models. The agent had to learn the negative consequences in an interactive fashion with a real building environment rather than exploring and learning it in an artificial manner with the help of the surrogate models. This is why $Ag_{RL,6}^{Imit+Guid}$ performs better than $Ag_{RL,5}^{Imit}$ during the training

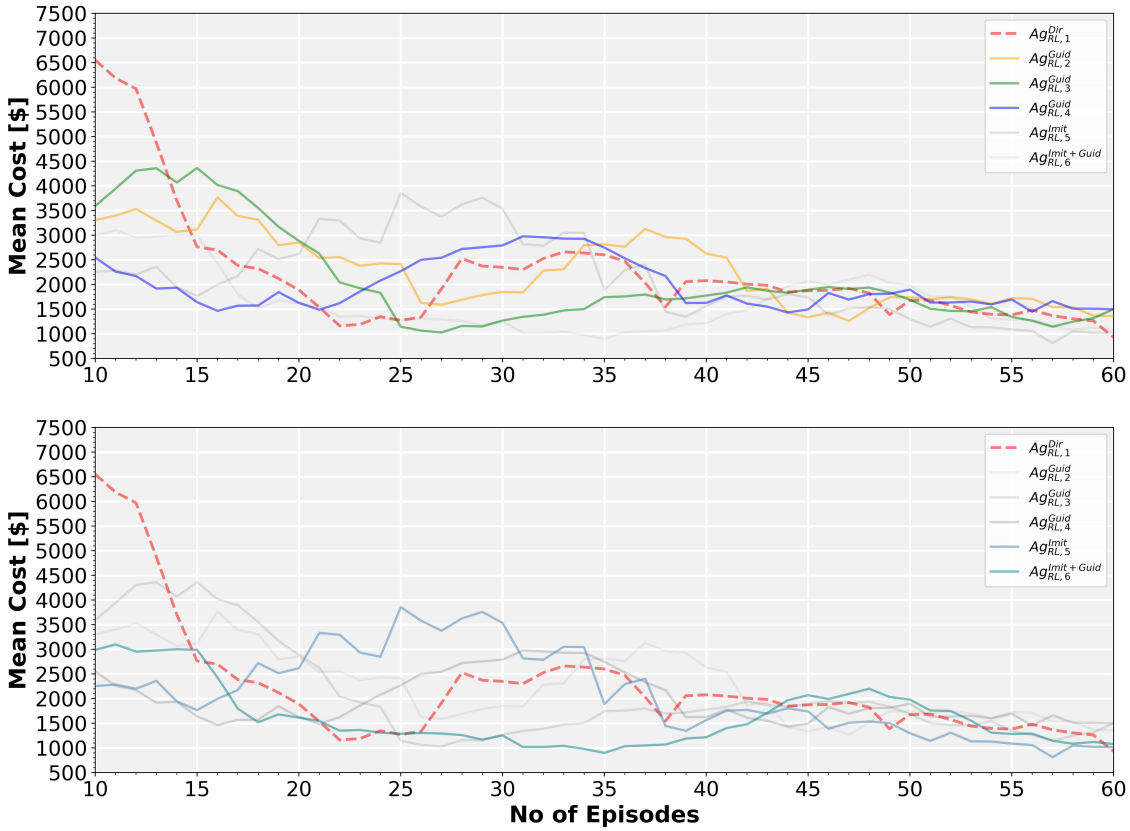

Fig. 12 Training progress of the different approaches adopted

period and is also more stable in its learning progress as shown in Figure 12. There is a rebound effect in the imitation learning approach as the agent blindly follows the rule-based control, and this is addressed with the guided surrogate model approach in Approach $Ag^{Imit+Guid}_{RL,6}$.

Approaches $Ag^{Guid}_{RL,3}$ and $Ag^{Guid}_{RL,4}$ are being trained on data generated from the artificial trajectories. Trajectories generated around rule-based controls in $Ag^{Guid}_{RL,4}$ had better prediction as rule-based data were already gathered from the interaction in this approach, whereas in approach $Ag^{Guid}_{RL,3}$, the artificial trajectories were generated randomly which had imperfect predictions. That is why, approach $Ag^{Guid}_{RL,4}$ had better performance than approach $Ag^{Guid}_{RL,3}$. Training on artificial and imperfect predictions from the trajectories generated by surrogate models yielded unsatisfactory training and testing performances with $Ag^{Guid}_{RL,3}$. $Ag^{Guid}_{RL,3}$ performed worst as the trajectories generated were fully random, and training on full random exploratory paths with imperfect predictions led to incorrect value estimates and prevented the agent from navigating toward the optimal action. Poor estimation of thermal behavior and reward functions with the surrogate building model and reward model, respectively, in the early stages leads to poor training behavior and harms the learning progress of the RL agent.

Limiting the exploration around a rule-based control strategy by controlling action paths within a certain bound led to better performance, as is evident from $Ag^{Guid}_{RL,4}$. We consider the rule-based actions to be the “safe" actions, and sticking to the rule-based actions preferentially in the early stages led to better estimation of the system and reward dynamics around the rule-based controls action space by the surrogate models. This led the RL agent to slowly nudge its policy away from the rule-based policy, where it expects better policy rewards, instead of shifting rapidly to an unseen state space, where it may have erroneous value estimates.

10 Future Work and Limitations

In order to further improve the efficacy of surrogate model training in future studies, a key focus should be on refining hyper-parameter settings. Conducting a systematic analysis to examine the impact of varying the number of generated sample trajectories, adjusting the look-ahead time, and optimizing the process for creating a lookup table that correlates states and actions with rewards, would be particularly useful. It is important to acknowledge the computational demands of this approach, as they significantly influence decisions regarding the extent of trajectory rollouts and the exploration depth at each step. However, utilizing high-performance computing (HPC) systems could enable the exploration of broader state spaces and more extended trajectories, potentially yielding improved outcomes, as evidenced by this research.

To fully identify the extent of performance enhancements attainable through expanded state space exploration, further research is required to define the upper limits of these improvements. Overall, continued investigation into hyper-parameter settings and exploration of state spaces will be crucial to further optimizing the surrogate model training process and achieving even better results in the future.

It is important to note that this approach is only applicable to off-policy learning algorithmsOff-policy learning algorithms are capable of learning and improving upon a policy that is different from the policy driving it’s current action choices. Surrogate models help in creating the data from the rule-based policy which is different from the existing policy of the RL agent that is interacting with the building systems. Nonetheless, utilizing building data in combination with surrogate models could be highly beneficial, potentially improving the performance and efficiency of the learning process. Future research could investigate the effectiveness of this approach in greater detail, exploring different weights for the short-term guided policy and further refining the learning process. Here we assumed that we have no prior building data to start with,

and the agent starts without any knowledge with the assistance of the surrogate models. Having building data at the initial stages will be quite beneficial and the same approaches can be utilized with less constraints on the artificial exploration process with the surrogate models.

The success of the guided exploration approach depends on short-term feedback rather than value functions that can capture the long-term consequences of the action. As a result, this approach may not be suitable for systems with very slow dynamics or those in which the response to a control action occurs delayed by several time steps later. In the context of building systems, the guided exploration approach may not be effective if shorter time-steps are taken, such as 30 seconds or one minute. In such scenarios, the time delay between control actions and their effect on the system may be too long to provide meaningful short-term feedback for the surrogate models. On the other hand, longer control-time steps of five to ten minutes are preferred, as they allow for a more sensible response to the control action executed in the building. In summary, the guided exploration approach may not be suitable for building systems with very short time steps, but can be effective when longer control-time steps are utilized, i.e., the approach is recommended when the dominant time constant of the system is shorter than the control time step.

Acknowledgment and Funding Data

The study is partially funded by the Building Energy Smart Technologies (BEST) Center, an industry research cooperative research center established under NSF Grants No. 213874 and No. 2113907, which the authors gratefully acknowledge. This work was supported in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding is provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Building Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

References

[1] for Buildings, G. A. and Construction, 2020, "Global Alliance for Buildings and Construction 2020 GLOBAL STATUS REPORT FOR BUILDINGS AND CONSTRUCTION, Towards a zero-emissions, efficient and resilient buildings and construction sector," http://www.un.org/Depts/

[2] Richter, B., Crabtree, G., Glicksman, L., Goldstein, D., Goldston, D., Greene, D., Kammen, D., Levine, M., Lubell, M., Savitz, M., and Sperling, D., 2008, "Energy Future: Think Efficiency," Unpublished, pp. 1–109.

[3] Tyra, B., Cassar, C., Liu, J., Wong, P., and Yildiz, O., 2019, "Electric Power Monthly with data for November 2020," .

[4] Chen, B., Cai, Z., and Bergés, M., 2019, "Gnu-RL: A precocial reinforcement learning solution for building HVAC control using a differentiable MPC policy," BuildSys 2019 - Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp. 316–325.

[5] Roth, A. and Reyna, J., 2019, "Grid-Interactive Efficient Buildings Technical Report Series: Whole-Building Controls, Sensors, Modeling, and Analytics," https://www.energy.gov/eere/buildings

[6] Wang, Z. and Hong, T., 2020, "Reinforcement learning for building controls: The opportunities and challenges," Applied Energy, **269**, p. 115036.

[7] Wang, Z. and Hong, T., 2020, "Reinforcement learning for building controls: The opportunities and challenges," Applied Energy, **269**(February), p. 115036.

[8] Kontes, G. D., Giannakis, G. I., Sánchez, V., de Agustin-Camacho, P., Romero-Amorrortu, A., Panagiotidou, N., Rovas, D. V., Steiger, S., Mutschler, C., and Gruen, G., 2018, "Simulation-based evaluation and optimization of control strategies in buildings," Energies, **11**.

[9] Andrew G Sutton, R. S. B., 2014, *Reinforcement Learning : An Introduction*, The MIT Press.

[10] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D., 2015, "Human-level control through deep reinforcement learning," Nature, **518**(7540), pp. 529–533.

[11] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. V. D., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D., 2016, "Mastering the game of Go with deep neural networks and tree search," Nature, **529**, pp. 484–489.

[12] Sallab, A. E., Abdou, M., Perot, E., and Yogamani, S., 2017, "Deep reinforcement learning framework for autonomous driving," arXiv preprint arXiv:1704.02532.

[13] Folkers, A., Rick, M., and Büskens, C., 2019, "Controlling an autonomous vehicle with deep reinforcement learning," *2019 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, pp. 2025–2031.

[14] Jebessa, E., Olana, K., Getachew, K., Isteefanos, S., and Mohd, T. K., 2022, "Analysis of Reinforcement Learning in Autonomous Vehicles," *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, IEEE, pp. 0087–0091.

[15] Levine, S., Finn, C., Darrell, T., and Abbeel, P., 2016, "End-to-end training of deep visuomotor policies," The Journal of Machine Learning Research, **17**(1), pp. 1334–1373.

[16] Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D., 2018, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," The International journal of robotics research, **37**(4-5), pp. 421–436.

[17] OpenAI, "ChatGPT: Optimizing Language Models for Dialogue," https://openai.com/blog/chatgpt/

[18] Dey, S., Marzullo, T., and Henze, G., 2023, "Inverse reinforcement learning control for building energy management," Energy and Buildings, **286**, p. 112941.

[19] Zhang, Z. and Lam, K. P., 2018, "Practical implementation and evaluation of deep reinforcement learning control for a radiant heating system," BuildSys 2018 - Proceedings of the 5th Conference on Systems for Built Environments, pp. 148–157.

[20] Breiman, L., 2001, "Random forests," Machine learning 45, pp. 5–32.

[21] Maimon, O. Z. and Rokach, L., 2014, *Data mining with decision trees: theory and applications*, Vol. 81, World scientific.

[22] Lecun, Y., Bengio, Y., and Hinton, G., 2015, "Deep learning," doi: 10.1038/nature14539.

[23] Henze, G. P. and Schoenmann, J., 2003, "Evaluation of reinforcement learning control for thermal energy storage systems," HVAC and R Research, **9**, pp. 259–275.

[24] Bertsekas, D. P., 2009, "Neuro-dynamic programming," Encyclopedia of Optimization, CA Floudas and PM Pardalos, Eds. Boston, MA: Springer US, pp. 2555–2560.

[25] Liu, S. and Henze, G. P., 2006, "Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory: Part 1. Theoretical foundation," Energy and Buildings, **38**, pp. 142–147.

[26] Liu, S. and Henze, G. P., 2006, "Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory: Part 2. Results and analysis," Energy and Buildings, **38**, pp. 148–161.

[27] Dalamagkidis, K., Kolokotsa, D., Kalaitzakis, K., and Stavrakakis, G. S., 2007, "Reinforcement learning for energy conservation and comfort in buildings," Building and Environment, **42**, pp. 2686–2698.

[28] Sutton, R. S., 1988, "Learning to Predict by the Methods of Temporal Differences," Machine learning, **3**, pp. 9–44.

[29] Yang, L., Nagy, Z., Goffin, P., and Schlueter, A., 2015, "Reinforcement learning for optimal control of low exergy buildings," Applied Energy, **156**, pp. 577–586.

[30] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., 2013, "Playing Atari with Deep Reinforcement Learning," arXiv preprint arXiv:1312.5602, pp. 1–9.

[31] Li, B. and Xia, L., 2015, "A multi-grid reinforcement learning method for energy conservation and comfort of HVAC in buildings," IEEE International Conference on Automation Science and Engineering, **2015-October**, pp. 444–449.

[32] Costanzo, G. T., Iacovella, S., Ruelens, F., Leurs, T., and Claessens, B. J., 2016, "Experimental analysis of data-driven control for a building heating system," Sustainable Energy, Grids and Networks, **6**, pp. 81–90.

[33] Wei, T., Wang, Y., and Zhu, Q., 2017, "Deep Reinforcement Learning for Building HVAC Control," doi: 10.1145/3061639.3062224.

[34] Li, Y., Wen, Y., Guan, K., and Tao, D., 2017, "Transforming cooling optimization for green data center via deep reinforcement learning," arXiv, pp. 1–11.

[35] Silver, D., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M., 2014, "Deterministic Policy Gradient Algorithms," .

[36] Murugesan, S., Jiang, Z., Risbeck, M. J., Amores, J., Zhang, C., Ramamurti, V., Drees, K. H., and Lee, Y. M., 2020, "Less is More: Simplified State-Action Space for Deep Reinforcement Learning based HVAC Control," RLEM 2020 - Proceedings of the 1st International Workshop on Reinforcement Learning for Energy Management in Buildings and Cities, pp. 20–23.

[37] Chen, B., Cai, Z., and Bergés, M., 2019, "Gnu-RL: A precocial reinforcement learning solution for building HVAC control using a differentiable MPC policy," BuildSys 2019 - Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp. 316–325.

[38] Amos, B., Rodriguez, I. D. J., Sacks, J., Boots, B., and Kolter, J. Z., 2018, "Differentiable MPC for End-to-end Planning and Control," http://arxiv.org/abs/1810.13400

[39] Zou, Z., Yu, X., and Ergan, S., 2020, "Towards optimal control of air handling units using deep reinforcement learning and recurrent neural network," Building and Environment, **168**(November 2019), p. 106535.

[40] Costanzo, G. T., Iacovella, S., Ruelens, F., Leurs, T., and Claessens, B. J., 2016, "Experimental analysis of data-driven control for a building heating system," Sustainable Energy, Grids and Networks, **6**, pp. 81–90.

[41] Arroyo, J., Manna, C., Spiessens, F., and Helsen, L., 2022, "Reinforced model predictive control (RL-MPC) for building energy management," Applied Energy, **309**.

[42] Spangher, L., Gokul, A., Khattar, M., Palakapilly, J., Agwan, U., Tawade, A., and Spanos, C., 2020, "Augmenting Reinforcement Learning with a Planning Model for Optimizing Energy Demand Response," RLEM 2020 - Proceedings of the 1st International Workshop on Reinforcement Learning for Energy Management in Buildings and Cities, pp. 39–42.

[43] Pillonetto, G., Aravkin, A., Gedon, D., Ljung, L., Ribeiro, A. H., and Schön, T. B., 2023, "Deep networks for system identification: a Survey," .

[44] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S., 2018, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 35th International Conference on Machine Learning, ICML 2018, **5**, pp. 2976–2989.

[45] Marzullo, T., Dey, S., Long, N., Vilaplana, J. L., and Henze, G., 2022, "A high-fidelity building performance simulation test bed for the development and evaluation of advanced controls," Journal of Building Performance Simulation, **15**, pp. 379–397.

[46] Dey, S., Marzullo, T., Zhang, X., and Henze, G., 2023, "Reinforcement learning building control approach harnessing imitation learning," Energy and AI, p. 100255.

[47] Deru, M., Field, K., Studer, D., Benne, K., Griffith, B., Torcellini, P., Liu, B., Halverson, M., Winiarski, D., Rosenberg, M., et al., 2011, "US Department of Energy commercial reference building models of the national building stock," .

[48] https://www.ziprecruiter.com/Salaries/-in-Chicago,IL

[49] Seppänen, O., Fisk, W. J., and Faulkner, D., 2006, "Cost Benefit Analysis of the Night-Time Ventilative Cooling in Office Building," Proceedings of Healthy Buildings 2006, pp. 243–247.

[50] Local, E., 2012, "Chicago, IL Electricity Rates | Electricity Local," accessed 2021-07-11, https://www.electricitylocal.com/states/illinois/chicago/

List of Figures

List of Tables