ORIGINAL RESEARCH **Open access**

Constraint-Decoupled Safe Reinforcement Learning for Autonomous Energy System Control

Victor Moreno[1*], Daniela Ruiz[2], Andres Salazar[1]

Abstract

Reinforcement learning (RL) represents an encouraging optimal control strategy for multi-energy management systems. Because it operates without an a priori model, RL diminishes both initial and continuous project-specific engineering demands while adeptly capturing enhanced representations of inherent system dynamics. Nevertheless, standard RL lacks guarantees for constraint satisfaction, which can lead to multiple hazardous interactions within the operational environment. This study introduces two innovative online, model-free safe RL techniques, termed SafeFallback and GiveSafe, which effectively separate the formulation of safety constraints from the RL architecture. These approaches ensure strict adherence to hard constraints throughout both the training phase and the execution of the (near-)optimal policy. Crucially, this is achieved without solving a mathematical program, thereby reducing computational overhead and allowing for greater flexibility in defining constraint functions. Through a simulated case study involving multi-energy systems, we demonstrate that both proposed methods initiate with substantially greater utility than both a standard RL baseline and an Optlayer baseline (94.6% and 82.8% versus 35.5% and 77.8%, respectively). Furthermore, the SafeFallback approach can even surpass the performance of the standard RL baseline (102.9% versus 100%). We conclude that these methodologies are robust safety-constraint mechanisms that extend well beyond RL applications, as evidenced by their ability to maintain hard-constraint guarantees even when employing random policies.

Keywords Reinforcement learning, Energy management system, Constraints, Multi-energy systems

*Correspondence:
Victor Moreno
victor.moreno@ub.edu
[1] Department of AI for Electrical Power Systems, Faculty of Engineering, University of Barcelona, Barcelona, Spain
[2] Department of Energy Informatics, School of Engineering, Autonomous University of Madrid, Madrid, Spain

Introduction

As technologies facilitating sector coupling reach maturity and see widespread deployment, energy systems are growing progressively more interconnected. This integration enables unified control strategies that can significantly improve the holistic efficiency and functionality of these multi-energy, multi-carrier, multi-commodity, or multi-utility frameworks. Moreover, such systems facilitate the exploitation of operational flexibility (e.g., controllable loads and storage) across and within distinct energy carriers. Generally, these integrated control mechanisms [1] are driven by economic, environmental, or combined objective functions, making them inherently multi-objective.

Achieving optimal or Pareto-optimal operational states in these multi-energy networks requires precise set-points to secure and sustain the target objectives (such as minimizing energy expenditures or CO2-equivalent emissions) while simultaneously satisfying all operational constraints [2]. Because flexibility deployment is highly dynamic and creates interdependencies across consecutive time intervals, it is essential to optimize over multiple time horizons. Furthermore,

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (CC BY-NC-SA 4.0), which permits non-commercial use, distribution, reproduction, and adaptation in any medium or format, provided that appropriate credit is given to the original author(s) and source, a link to the license is provided, and any changes made are indicated. Adapted material must be distributed under the same license.

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

maintaining the stability of the multi-energy system necessitates the careful management of various uncertainties, including fluctuations in pricing, weather conditions, and energy demand.

Recent studies have evaluated both model-predictive control (MPC) and reinforcement learning (RL) within the framework of multi-energy management [3]. Research by Ceusters *et al*. [3] demonstrated that RL-driven energy management systems do not depend on an initial model and can surpass linear MPC frameworks post-training. Despite this, standard RL **(Figure 1a)** executes numerous potentially hazardous exploratory actions in its environment, a trait that is impermissible for many practical applications. In multi-energy systems, for instance, failing to respect fundamental energy balance constraints could lead to severe under- or over-production. For most power networks, such imbalances might trigger breaches of maximum grid power capacity limits—a critical concern given the anticipated widespread adoption of electric vehicles. Furthermore, these imbalances pose acute risks for isolated energy setups disconnected from centralized district heating or larger distribution grids, which lack supplementary oversight. In those scenarios, imbalances frequently degrade end-user comfort, manifesting as heating failures or power blackouts.

Consequently, the objective of this research is to guarantee that all interactions with the target environment (specifically, a multi-energy system in our analysis) comply with a predefined set of safety constraints, regardless of the distinct (optimal) control methodology applied **(Figure 1b)**. By pursuing this architectural decoupling rather than engineering a dedicated safe RL algorithm, we ensure that subsequent, potentially superior optimization models can be seamlessly integrated in the future.

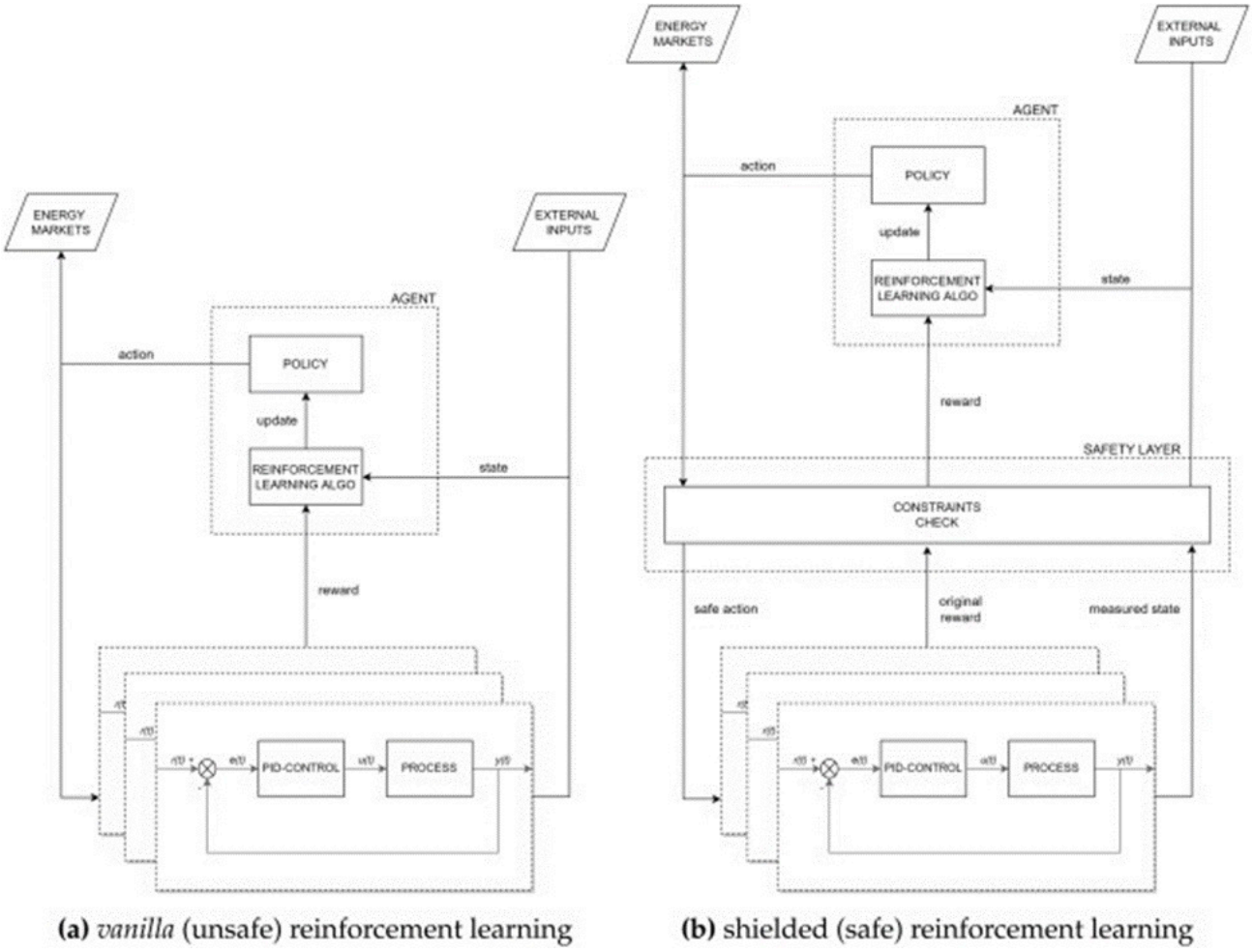


Figure 1. Comparison of block diagrams: the viability of the actions proposed by the RL agent, given a specific state, is continuously evaluated against pre-established constraint functions that serve as a safety mechanism, thereby protecting

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

the system from dangerous (control) interventions. It is assumed here that ongoing unconstrained error management (namely, reducing the variance between a monitored process variable and the target set-point) is handled by proportional-integral-derivative (PID) controllers.

Contribution and outline

To the best of our understanding, the primary contributions of this work include:

· An online, model-free safe RL technique that ensures strict hard-constraint satisfaction—as opposed to soft or chance constraints—while yielding notably superior initial utility;

· An architectural decoupling of the safety constraint definitions from the underlying RL framework, enabling the effortless substitution of future, theoretically superior optimization algorithms;

· The achievement of hard-constraint guarantees without relying on the solution of a mathematical program, which consequently lowers computational demands and affords greater adaptability in formulating constraint functions (as derivative data is unnecessary);

· The practical demonstration of an energy management system governed by safe RL within a comprehensive multi-energy simulation platform.

Related Work

Vanilla (unsafe) RL in multi-energy systems

Over the past few years, a multitude of studies have suggested applying RL across a variety of energy system applications. This body of research has been systematically reviewed by scholars such as Cao *et al*. [4], Yang *et al*. [5], and Perera and Kamalaruban [6], and further contextualized for the notably complex domain of multi-energy systems by Zhou [7]. Most of these use cases fall within the wider scope of energy management. In the context of multi-energy systems, practical demonstrations of RL-driven energy management frameworks include the pioneering work of Rayati *et al*. [8]. They utilized Q-learning [9] to manage energy within a simulated residential multi-energy structure, later incorporating demand-side management features [10]. Subsequently, Mbuwir *et al*. [11] successfully implemented RL (Q-learning) for a battery management setup within a comparable simulated residential environment, incorporating a fallback policy to override the RL agent whenever constraints were breached. Additionally, Wang *et al.* [12] combined a path-tracking interior point technique with an RL algorithm (Q-learning) to develop a bi-level interactive decision-making framework involving multiple agents in a regional multi-energy context. Ahrarinouri *et al*. [13] similarly introduced a multi-agent RL (Q-learning) methodology for energy management in simulated residential multi-energy systems. Concurrently, Ye *et al.* [14] applied a deep RL algorithm, specifically a deep deterministic policy gradient (DDPG) [15], enhanced with a prioritized experience replay mechanism, to a simulated residential multi-energy setup.

Furthermore, Xu *et al*. [16] presented a scheduling framework for industrial multi-energy systems that employs an RL (Q-learning)- based differential evolution strategy to dynamically determine the most effective mutation mechanism and its corresponding parameters. Along similar lines, Zhu *et a*l. [17] implemented an energy management system powered by multi-agent counterfactual soft actor-critic (mCSAC) [18] deep RL for a simulated multi-energy industrial complex. Zhou *et al*. [19] introduced a data-driven stochastic energy management approach utilizing deep reinforcement learning, integrating prioritized experience replay to bolster training efficiency and algorithm convergence. Additionally, Ceusters *et al*. [3] formulated both on-policy and off-policy model-free RL methodologies for multi-objective scenarios. Utilizing proximal policy optimization (PPO [20]) and twin delayed deep deterministic policy gradient (TD3 [21]), they evaluated these against a linear MPC framework, with both approaches rooted in foundational optimal control theory. Their findings

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

across two distinct simulated multi-energy environments indicated that RL agents can equal or exceed the performance of MPC. Meanwhile, Zhang *et al*. authored a sequence of studies [22-24] focusing on the (near-)optimal scheduling of an integrated energy system (i.e., multi-energy system) via deep reinforcement learning for both single- and multi-objective scenarios, which Zhang *et al*. [25] subsequently broadened to encompass distributed multi-energy systems via multi-agent deep reinforcement learning.

Safe RL in multi-energy systems

By its very nature, RL relies on interacting with an environment. It is crucial to implement sufficient safeguards to prevent the breach of environment-specific constraints, not only during active online training but also during the pure execution of a finalized policy (such as post-offline training). Whether explicitly acknowledged or not, the studies largely omitted or severely downplayed these specific environmental restrictions, thereby constraining their applicability in real-world scenarios. Breaching operational constraints in practical settings can trigger severe negative consequences, including financial deficits, compromised comfort, and potentially, physical injury to humans. Consequently, safe RL is designed to: “learn policies that maximize the expectation of the return in problems in which it is important to ensure reasonable system performance and/or respect safety constraints during the learning and/or deployment processes,” according to the definition provided by García and Fernández [26].

Through an extensive review extending beyond energy management, García and Fernández [26] evaluated the broader applications of safe RL, categorizing the methodologies into two primary strategies: (1) adjusting the optimality criteria by integrating a safety margin, and (2) altering the exploration phase by utilizing external knowledge or risk-based metrics for guidance. More recently, Dulac-Arnold *et al*. [27] outlined nine critical obstacles that must be surmounted to deploy RL in practical, real-world contexts, specifically highlighting the risk of safety constraint violations—a concern echoed by Nweye *et al*. [28]. Additionally, Brunke *et al.* [29] conducted a comprehensive examination of safe learning spanning both control theory and RL domains. They specifically detailed: (1) control techniques based on learning that begin with an a priori model to refine policies amidst dynamic system uncertainties securely; (2) safe RL methods that require neither an upfront model nor predefined constraints, though they lack definitive safety guarantees (despite promoting operational safety and robustness); and (3) methodologies that issue safety certifications for any acquired control policy.

As an early endeavor to merge hard-constraint adherence with RL within energy infrastructures, Venayagamoorthy *et al.* [30] introduced an intelligent dynamic energy management framework for a smart microgrid. This system utilized an action-dependent heuristic dynamic programming model that functions as an adaptive critic-based controller. To iteratively enhance the dispatch solution, they incorporated an evolutionary algorithm, systematically discarding any proposed candidate actions that failed the critical load fulfillment requirement, based on established power-balancing protocols and a rudimentary decision-tree management scheme. Additionally, Zhang *et al*. [31] conceptualized a bi-level power management architecture for interconnected microgrids operating within electrical distribution networks. At the primary tier, a cooperative agent leverages an adaptive, model-free RL algorithm to determine the optimal retail price indicators for the microgrids. At the secondary tier, individual model-based microgrid controllers execute a constrained mixed-integer nonlinear program informed by the price data supplied by the RL agent. Similarly, Zhao *et al.* [32] developed a knowledge-augmented RL structure that pairs a low-fidelity analytical model with an RL agent to manage cooperative wind farm controls. Should the RL agent select a naive action, a constrained alternative is computed by running an optimization process against the analytical model. Nonetheless, all these methodologies remain heavily dependent on prior physical models integrated into secondary optimization routines. Such architectures inherit a hypothesized transition function and an isolated objective function (distinct from the primary reward signal, potentially introducing bias) alongside constraint parameters—rather than relying exclusively on constraint functions.

Recently, and independently of our research, Park *et al*. [33] created an innovative RL algorithm heavily influenced by OptLayer [34], termed distance-based incentive/penalty Q-learning (DIP-QL). Demonstrated on a microgrid control scenario, this method similarly circumvents the need for an a priori transition function, relying exclusively on constraint parameters to establish strict hard-constraint assurances. However, their proposed technique fundamentally depends on

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

a deep Q-learning algorithm as its backbone. In contrast, our approach advocates for the complete decoupling of constraint management from the core RL algorithm, enabling the seamless integration of subsequent, theoretically superior optimization models, as our framework is not restricted solely to RL applications.

Conclusions from related work

An analysis of existing literature reveals that RL is a highly promising and frequently recommended approach for a broad spectrum of energy system applications (along with other fields not addressed herein), particularly concerning energy management systems. Furthermore, transitioning RL toward real-world deployment is highly complex and demands meticulous focus on operational safety. Various safe (reinforcement) learning methodologies are available, categorized by their safety assurances (from weakest to strongest): (1) soft-constraint satisfaction, (2) chance-constraint satisfaction, and (3) hard-constraint satisfaction. To the best of our awareness, however, no prior work has introduced and validated a model-free safe RL approach that seamlessly integrates all of the following attributes: (i) the provision of hard-constraint guarantees—superseding both soft- and chance-constraint compliances—across multiple variables; (ii) total architectural separation from the RL (modeled as a Markov Decision Process) formulation; (iii) functional reliability throughout the training of a (near) optimal policy (accommodating both greedy exploitation and exploratory actions) as well as during the deployment phase of any generalized policy (such as offline trained RL agents or random actors); (iv) the avoidance of having to solve a mathematical program; and (v) empirical demonstration within the context of multi-energy system energy management.

Materials and Methods

Adhering to the conventional RL definition of the state-value function while expanding it to subject it to constraints, the primary objective is to determine a policy π, which functions as a mapping of states, $s \in S$, to actions, $a \in (s)$,v that maximizes an expected sum of discounted rewards and is subject to constraint sets X and U:

$$\max_{\pi}\left(E_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^{k}(R-c)_{t+k+1} \right\} \right) \quad (1a)$$

$$s.t. \quad s_t = s \qquad t \in \mathbb{T}_0^{+\infty} = 0, \ldots, +\infty \quad (1b)$$

$$s_t \in X \qquad t \in \mathbb{T}_0^{+\infty} = 0, \ldots, +\infty \quad (1c)$$

$$a_t \in U \qquad t \in \mathbb{T}_0^{+\infty} = 0, \ldots, +\infty \quad (1d)$$

Where Eπ represents the expected value, executed under policy π, concerning the associated rewards R and costs c (which is explicitly delineated here to clarify negative reward penalties triggered when suggested actions breach predefined constraints)—adjusted by the discount factor γ across an infinite sum at any given time step t. The precise computational procedures are detailed in **Algorithms 1 and 2.**

It should be noted that **Eq. 1a** represents a discrete1 time-invariant, infinite-horizon stochastic optimal control problem, consistent with the observations of Ceusters *et al*. [3]. However, it diverges from conventional RL paradigms due to the inclusion of a priori constraint functions embedded within sets X and U. While techniques lacking a priori constraint parameters do exist, the current state-of-the-art for such methods maxes out at safety level 2 (chance-constraint satisfaction). In contrast, our approach ensures safety level 3 (hard-constraint satisfaction), adhering to the classification by Brunke *et al*. [29]. Instead of designing a singular, highly specialized safe RL algorithm, we advocate for dissociating

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

the a priori constraint function architecture from the primary (RL) agent. This separation ensures that any (new RL) algorithm can be utilized—while consistently guaranteeing hard-constraint satisfaction. Even though the computational logic of these proposed algorithms remains theoretically straightforward, their practical efficacy will be subsequently demonstrated on a multi-energy system (specifically incorporating a non-grid-connected thermal system).

SafeFallback method

Our initial proposed approach depends on a predefined safe fallback policy, πsafe. This policy can generally be established using conventional control theory, often consisting of rigid, hard-coded rules, such as a straightforward rule-based framework. Additionally, assuming the constraint functions are already defined, we can easily verify whether the chosen actions a taken in state s comply with the necessary constraint parameters. If these conditions are met, the selected actions a are classified as safe actions. These are subsequently implemented within the environment, yielding observations of the ensuing state s′, the reward r, and the done signal d. This sequence forms the standard experience tuple (s, $asafe$, r, s', d) utilized by the RL agent. Conversely, should the chosen action a fail the constraint verification, it is superseded by the safe action $asafe$ dictated by the predefined safe fallback policy πsafe. In this scenario, two distinct experience tuples are generated: the standard tuple (s, $asafe$, r, s', d) and an additional tuple (s, a, r - c, s', d), which records the invalid action alongside an associated negative reward (or cost, c). **Algorithm 1** outlines the corresponding pseudo-code.

Input: initialize RL algorithm, initialize constraint functions in sets X and U, initialize safe fallback policy π^{safe}
for $k = 0, 1, 2, \ldots$ **do**
  Observe state s and select action a
  if *constraint check* = *True* **then**
    keep selected action a as safe action a^{safe}
  else
    get safe action a^{safe} from safe fallback policy π^{safe}
  end
  Execute a^{safe} in the environment
  Observe next state s', reward r and done signal d to indicate whether s' is terminal
  Give experience tuple (s, a^{safe}, r, s', d) **and if** $a^{safe} \neq a : (s, a, r - c, s', d)$ with cost c
  If s' is terminal, reset environment state
end

Algorithm1. SafeFallback

GiveSafe method

The second approach we introduce eliminates the need for a predetermined safe fallback policy. Instead, it relies entirely on the RL agent itself to output safe actions $asafe$, which are continuously evaluated against the established constraint conditions. Should the chosen actions a in state s successfully clear the constraint verification, these safe actions $asafe$ are deployed within the environment, generating the conventional experience tuple (s, $asafe$, r, s', d). However, if the constraints are breached, the RL agent is presented with the experience tuple (s, a, c, s, d). Consequently, because the invalid action is not implemented, the system does not advance to the subsequent state, and a cost c (representing a negative reward) is applied. Following this, the RL agent must formulate a new action a, prompting another constraint evaluation. This loop continues until an action successfully clears the verification process, designating it as a safe action $asafe$. This verified action is then enacted in the environment, yielding a standard experience tuple. **Algorithm 2** details the relevant pseudo-code, while **Figure 2** illustrates this process conceptually as a Markov Chain.

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

Input: initialize RL algorithm, initialize constraint functions in sets X and U
for $k = 0, 1, 2, \ldots$ **do**
  Observe state s and select action a
  if *constraint check* = *True* **then**
    keep selected action a as safe action a^{safe}
  else
    while *constraint check* = *False* **do**
      give experience tuple (s, a, c, s, d) with cost c
      agent selects new action a
      check constraints
    end
    return safe action a^{safe}
  end
  Execute a^{safe} in the environment
  Observe next state s', reward r and done signal d to indicate whether s' is terminal
  Give experience tuple (s, a^{safe}, r, s', d)
  If s' is terminal, reset environment state
end

Algorithm 2. GiveSafe

It is important to note that within this framework, the cost c should ideally be structured as an informative function (a shaped cost), given that the agent gains no advantage from a shaped reward if the constraint verification fails.

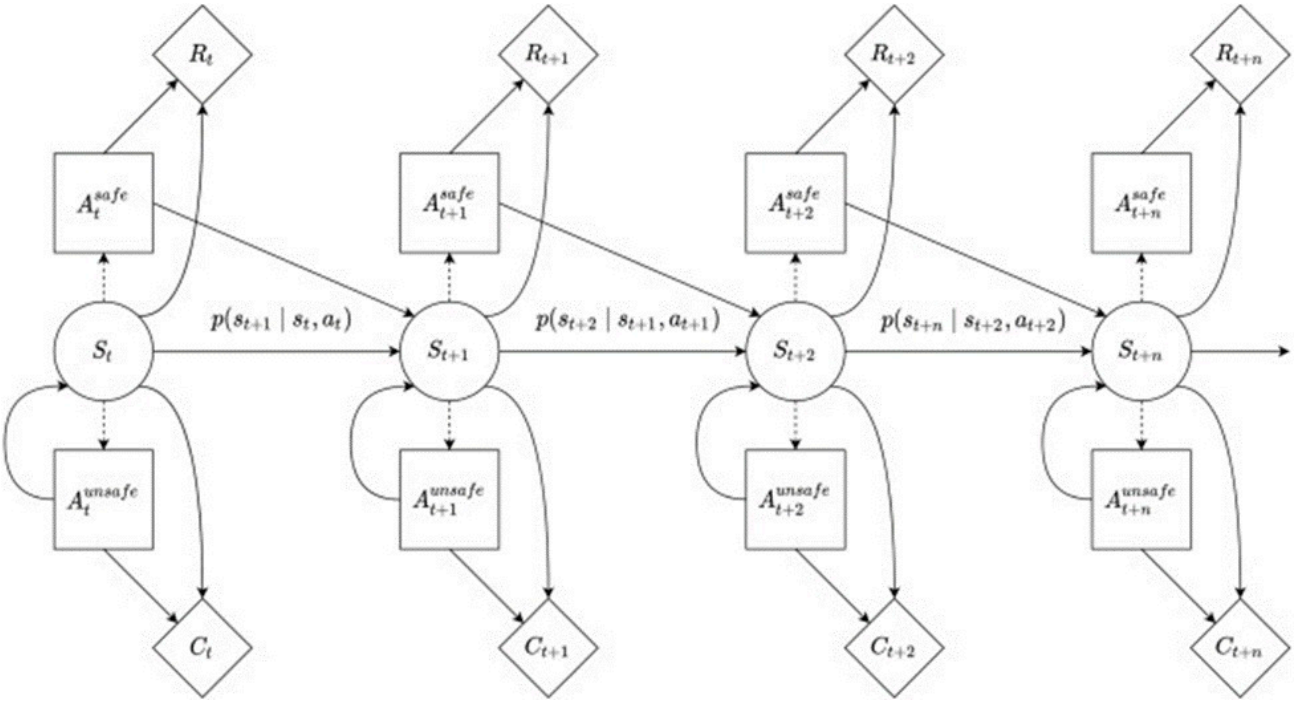


Figure 2. Markov Chain of **Algorithm 2**. When the selected action is infeasible (does not satisfy the constraints), that unsafe action A_t^{unsafe} is not executed in the environment, so no transition to the next state St+1 is observed, and a cost

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

Ct is given. When the selected actions are feasible (satisfy the constraints), those safe actions A_t^{safe} are executed in the environment, so a transition to the next state St+1 is observed with probability $p(st+1 \mid st, at)$ and a reward Rt is given.

Case Study

Toolchain

We employed a multi-energy system simulation model created by Ceusters *et al*. [3]. This model was selected because it permits the safe testing of critical operational parameters (such as verifying that all energy requirements are met) without the adverse consequences or dangers associated with breaching real-world constraints. The simulation was built using Modelica [35], which facilitates straightforward modeling of expected real-world system behaviors through multiphysics, first-principles equations, complemented by its object-oriented design, fundamental components, and access to highly specialized libraries.

Subsequently, this Modelica model was exported as a co-simulation functional mock-up unit (FMU), reflecting the approach in Ref. [36]. It was then integrated into a Python-based OpenAI gym environment [37], mirroring the methodology detailed in Refs. [3] and [38]. **Figure 3** displays the architectural layout of this toolchain. It should be noted that the Differential Algebraic Equations solver is embedded within the co-simulation FMU. Furthermore, the do_step() function from PyFMI [39] was utilized instead of simulate. As observed in Ref. [3], this choice provides a substantial enhancement in processing speed when properly initialized.

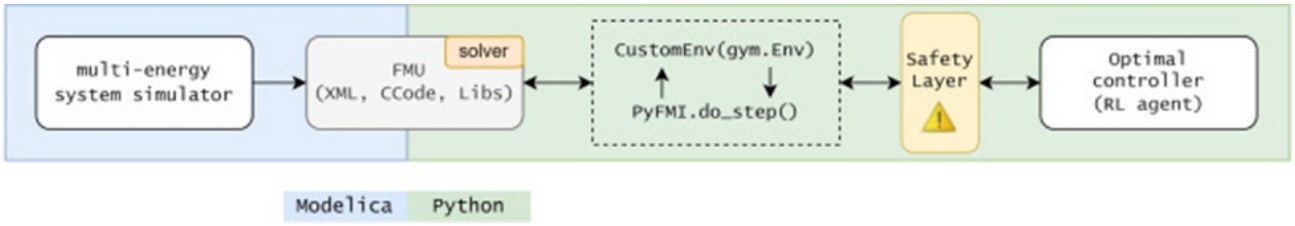


Figure 3. Architecture of the toolchain.

Simulation model

The multi-energy framework examined in this study closely resembles the architecture presented by Ceusters *et al*. [3], except that the gas turbine (backup generator) is omitted. The structural configuration is as follows **(Figure 4):**

Proceeding from left to right and top to bottom, the system incorporates: an electrical transformer, a wind turbine, a photovoltaic (PV) array, a natural gas boiler, a heat pump (HP), a combined heat and power (CHP) generator, a thermal energy storage system (TESS), and a battery energy storage system (BESS). The physical dimensions and capacities of this multi-energy network are also derived from [3] and are outlined in **Table 1.**

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

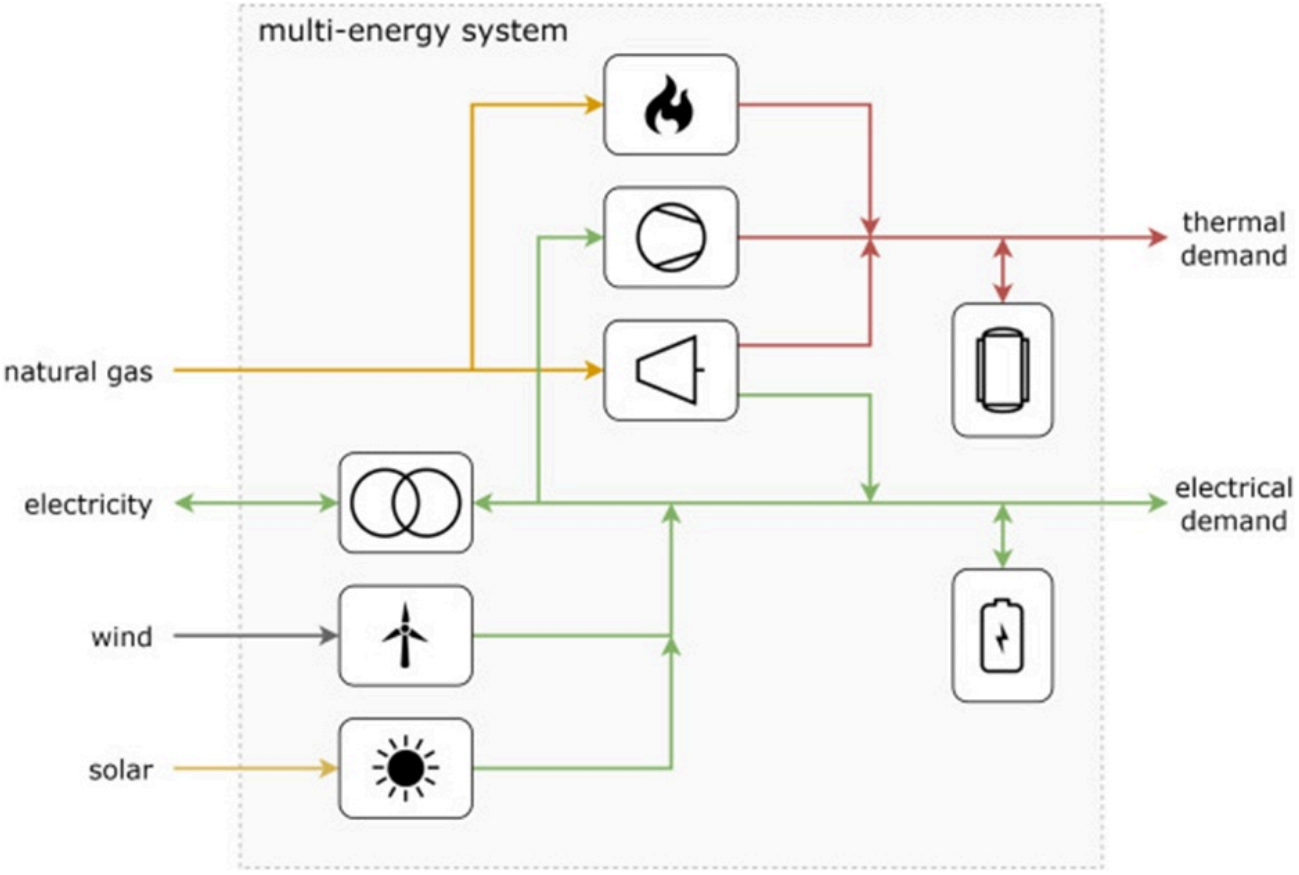


Figure 4. Structure of the simulated multi-energy system.

Although the dynamic simulation comprises a highly intricate set of differential-algebraic equations (totaling 2.548 equations with a corresponding number of variables [3]), it omits any simulated regulatory control mechanisms (such as the PID controllers illustrated in **Figure 1**). For this case study, we abstract this control layer. Consequently, any actions initiated within the simulation remain static throughout the designated control horizon (15 min), and continuous unconstrained error correction (the ongoing reduction of the discrepancy between a target set-point and an observed process variable) is excluded. We consider this abstraction reasonable—and it is consistently applied across all evaluated energy management algorithms to ensure fair comparison—though it introduces a control deviation of roughly 5%. This margin of error was identified through independent simulations running on a shortened 5 s control horizon.

Table 1. Dimensions of the multi-energy system.

Energy asset	Input	Output	E_{nom}	P_{min}	P_{nom}
Transformer	elec	elec		-∞	+∞
Wind turbine	wind	elec		1.5%	0.8 MW_e
Solar PV	solar	elec		0%	1.0 MW_e
Boiler	CH_4	heat		10%	2.0 MW_{th}
Heat pump	elec	heat		25%	1.0 MW_{th}
CHP	CH_4	heat		50%	1.0 MW_{th}
Empty Cell	CH_4	elec		50%	0.8 MW_e

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

TESS	heat	heat	3.5 MWh	−0.5 MW_{th}	+ 0.5 MW_{th}
BESS	elec	elec	2.0 MWh	−0.5 MW_e	+ 0.5 MW_e

Safety layer

In this specific case study, the constraint functions defined in **Eqs. 1c, 1d** take the following form (refer to the note regarding modelling effort associated with **Eq. 3c:**

$$Q_{boil}^{min} \times \gamma_{boil}^{t} \leq Q_{boil}^{t} \leq Q_{boil}^{max} \times \gamma_{boil}^{t} \qquad \forall t, \ \gamma_{boil}^{t} \in \{0,1\} \quad \text{(2a)}$$

$$\begin{bmatrix} Q_{hp}^{min} \\ P_{hp}^{min} \end{bmatrix} \times \gamma_{hp}^{t} \leq \begin{bmatrix} Q_{hp}^{t} \\ P_{hp}^{t} \end{bmatrix} \leq \begin{bmatrix} Q_{hp}^{max} \\ P_{hp}^{max} \end{bmatrix} \times \gamma_{hp}^{t} \qquad \forall t, \ \gamma_{hp}^{t} \in \{0,1\} \quad \text{(2b)}$$

$$\begin{bmatrix} Q_{chp}^{min} \\ P_{chp}^{min} \end{bmatrix} \times \gamma_{chp}^{t} \leq \begin{bmatrix} Q_{chp}^{t} \\ P_{chp}^{t} \end{bmatrix} \leq \begin{bmatrix} Q_{chp}^{max} \\ P_{chp}^{max} \end{bmatrix} \times \gamma_{chp}^{t} \qquad \forall t, \ \gamma_{chp}^{t} \in \{0,1\} \quad \text{(2c)}$$

$$Q_{tess}^{min} \leq Q_{tess}^{t} \leq Q_{tess}^{max} \qquad \forall t \quad \text{(2d)}$$

$$P_{bess}^{min} \leq P_{bess}^{t} \leq P_{bess}^{max} \qquad \forall t \quad \text{(2e)}$$

$$Q_{production}^{t} = Q_{demand}^{t} \qquad \forall t \quad \text{(2f)}$$

Here,Q_{boil}^{t} , Q_{hp}^{t},Q_{chp}^{t} and Q_{tess}^{t} signify the thermal power outputs of the natural gas boiler, heat pump, combined heat and power (CHP) unit, and thermal energy storage system (TESS), respectively. Similarly, P_{hp}^{t} ,P_{chp}^{t} and P_{bess}^{t} indicate the electrical power consumption or output for the heat pump, CHP, and battery energy storage system (BESS). All of these are bounded by their respective minimum and maximum capacity limits **(Table 1)**. Additionally, γ_{boil}^{t}, γ_{hp}^{t} and γ_{chp}^{t} function as binary indicators used to toggle the operational state of these assets (necessary because their minimum operating powers exceed zero). As a result, they do not necessitate an independent constraint verification process.

Assuming the electrical grid connection is robust enough to handle capacity demands, the electrical energy balance is continuously satisfied. In contrast, ensuring strict adherence to the thermal energy balance **(Eq. 2f)** requires dedicated oversight to guarantee hard-constraint compliance. We exclude any supplementary operational constraints in this analysis (such as ramp-up/ramp-down limits, or mandated minimum run/idle times), since energy balance conditions are generally recognized as the primary limiting factors in energy management scenarios. Expanding the thermal energy balance and easing the strict equality constraint (converting it into an inequality constraint) yields:

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

$$\left|Q^t_{boil} + Q^t_{hp} + Q^t_{chp} + Q^t_{tess} - Q^t_{demand}\right| \leq Q_{tol} \quad \forall t \tag{3a}$$

$$\left|A^t_{boil} \cdot \eta_{boil} \cdot Q^{max}_{boil} + A^t_{hp} \cdot \frac{COP_{hp}}{COP^{max}_{hp}} \cdot Q^{max}_{hp} + A^t_{chp} \cdot \eta^{th}_{chp} \cdot Q^{max}_{chp} + A^t_{tess} \cdot f(SOC^t_{tess}) - Q^t_{demand}\right| \leq Q_{tol} \quad \forall t \tag{3b}$$

$$\left|A^t_{boil} \cdot f(T^t_{boil}) \cdot Q^{max}_{boil} + A^t_{hp} \cdot \frac{f(T^t_{evap}, T^t_{cond})}{COP^{max}_{hp}} \cdot Q^{max}_{hp} + A^t_{chp} \cdot f(P^t_{chp}, Q^t_{chp}, T^t_{env}) \cdot Q^{max}_{chp} + A^t_{tess} \cdot f(\overline{T^t_{tess}}) - Q^t_{demand}\right| \leq Q_{tol} \quad \forall t \tag{3c}$$

In this context, At represents the (control) actions, η stands for energy efficiencies, COP denotes the coefficient of performance, SOC indicates the state of charge, and T refers to an array of specific thermal metrics (specifically, T^t_{boil} for the boiler return temperature, T^t_{evap} for the heat pump evaporator temperature, T^t_{cond} for the heat pump condenser temperature, T^t_{env} for ambient air temperature, and T^t_{tess} for the mean temperature within the layered hot water storage tank). The tolerance parameter Q_{tol} is established at 15,0% of the aggregate Q^t_{demand} during the assessment timeframe. While we recognize this percentage is somewhat elevated, it was selected to restrict computational overhead. It is important to emphasize that deriving precise models for the various functions f (.) in **Eq. 3c** is generally complex. Consequently, we presume access to a historical data archive, allowing us to apply supervised learning (specifically, a Random Forest Regression technique) to map the relationship between thermal power and applied actions directly (i.e. Q^t_{asset} = *f* (A^t_{asset}, X^t_{asset})). This method accommodates the integration of relevant external variables X_asset^t. To accurately reflect practical real-world applications, this training dataset is produced via the simulation environment rather than extracted directly from the fundamental differential algebraic equations.

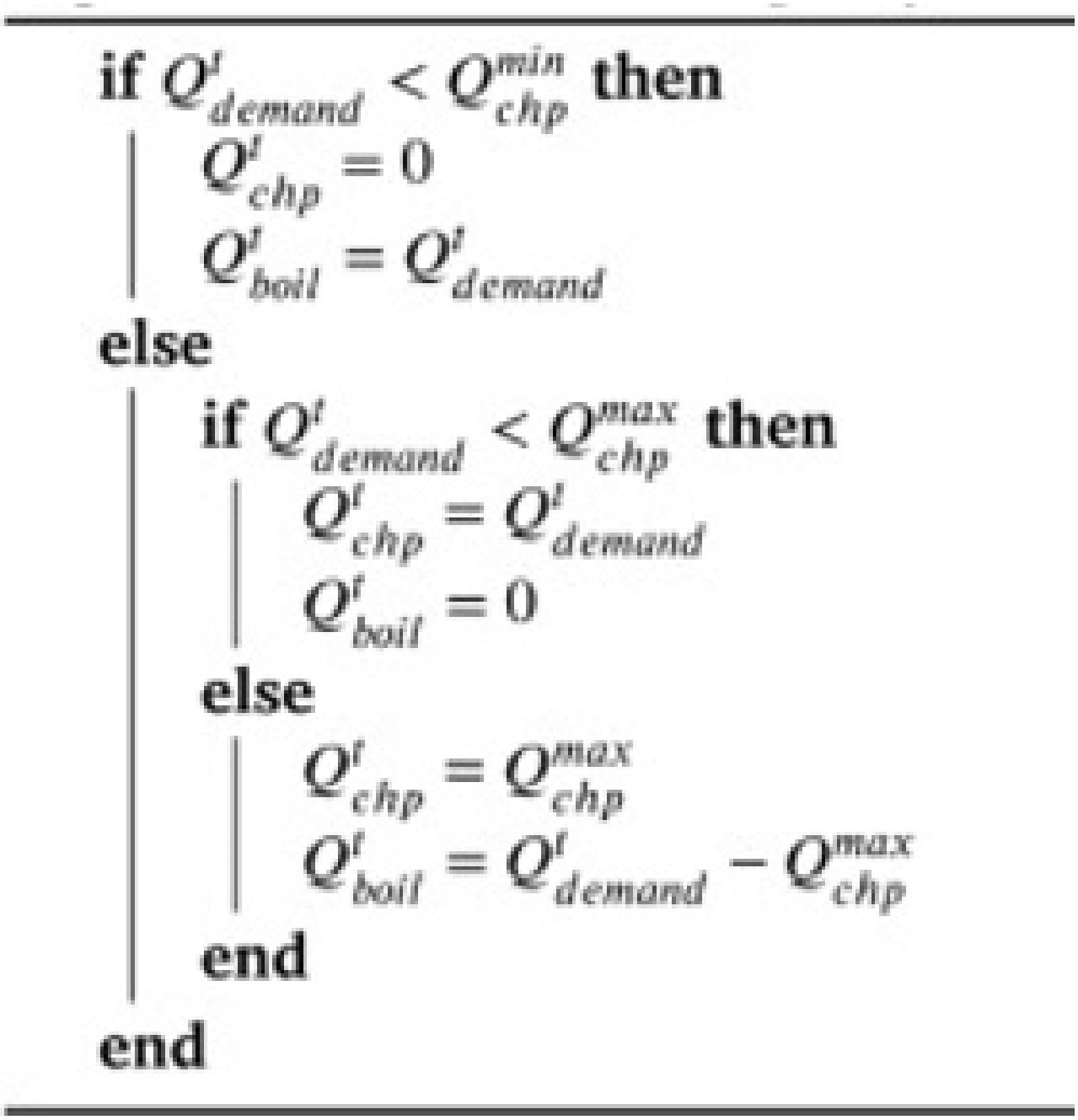
Table 2. Safety layer model metrics with test_size of 0.25. Mean Absolute Error (MAE), Normalized Mean Absolute Error (NMAE) by range, i.e., NMAE = MAE/range(actual values).

Energy asset	NMAE	MAE	R2-score
Boiler	0.34%	7.2 kW	99.92%
Heat pump	0.62%	6.1 kW	99.74%
CHP	0.38%	4.3 kW	99.86%
TESS	1.37%	12.9 kW	96.22%
BESS	0.46%	4.6 kW	99.43%

Safe fallback policy

As detailed, **Algorithm 1** depends on a predetermined safe fallback policy πsafe. This can take the form of any feasible, albeit potentially suboptimal, strategy that adheres to the system constraints and is generally formulated by subject matter experts. Within the context of our case study, this manifests as a straightforward priority rule.

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press



Algorithm 3. Safe fallback policy

It should be noted that, for the sake of clarity, this policy is articulated using thermal power outputs; however, it is ultimately translated into the specific control actions A^t_{chp} and A^t_{boil} during the transition from **Eq. 3a** to **Eq. 3b.**

Energy-managing RL agent

The fully observable discrete-time Markov decision process (MDP) is structured as the tuple $\langle S, A, Pa, Ra \rangle$, defined as follows:

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

$$S^t = (E^t_{th},\ E^t_{el},\ P^t_{wind},\ P^t_{solar},\ X^t_{el},\ SOC^t_{tess},\ SOC^t_{bess},\ h^t,\ d^t) \qquad S^t \in S \tag{4a}$$

$$A^t_{boil} = (0,\ A^{min}_{boil} \rightarrow A^{max}_{boil}) \qquad A^t_{boil} \in A \tag{4b}$$

$$A^t_{hp} = (0,\ A^{min}_{hp} \rightarrow A^{max}_{hp}) \qquad A^t_{hp} \in A \tag{4c}$$

$$A^t_{chp} = (0,\ A^{min}_{chp} \rightarrow A^{max}_{chp}) \qquad A^t_{chp} \in A \tag{4d}$$

Here, E^t_{th} represents the required thermal energy, E^t_{el} signifies the required electrical energy, P^t_{wind} denotes the electrical power supplied by the wind turbine, P^t_{solar} indicates the electrical power generated by the solar panels, X^t_{el} is the market price for electricity (specifically, the day-ahead spot market price), SOC^t_{tess} defines the state-of-charge for the TESS, SOC^t_{bess} is the state-of-charge for the BESS, ht stands for the current hour, and dt indicates the current day of the week. All these variables are captured at the tth interval, collectively forming the state-space S. The corresponding action-space A encompasses the operational set-points for the various assets: A^t_{boil} for the natural gas boiler, A^t_{hp} for the heat pump, A^t_{chp} for the CHP system, A^t_{tess} for the TESS, and A^t_{bess} for the BESS. These set-points are strictly bounded by their specific minimum and maximum power constraints, as detailed in **Table 1.**

The primary objective of the energy management agent is dictated by the reward function Ra (in this context, aiming to maximize a negative value, effectively minimizing its positive counterpart, aligning with **Eq. 1a**). This objective incorporates the negative impact of energy expenses L^t_{cost} and comfort deterioration $L^t_{comfort}$ at the tth time-step, scaled by the respective weights a and b. Additionally, a penalty cost c is imposed whenever constraint violations are anticipated.3 Comfort deterioration is mathematically represented as $|E^t_{th} - Q^t|$, where Q_t stands for the total thermal energy generated. Within the Modelica framework, it is presumed that both electrical power and natural gas supplies can be fully satisfied via robust connections to infinitely large external utility grids. It is important to note that the comfort penalty $L^t_{comfort}$ remains restricted by the tolerance bounds set in **Eq. 3c.** Consequently, this component of the reward function acts as a precise calibration tool. It helps further reduce comfort loss within allowable limits (steering the RL agents toward more reliable operational choices) and compensates for any inherent inaccuracies within the constraint models themselves **(Table 2).**

Financial energy expenses L^t_{cost} are calculated in EUR, utilizing a scalarization multiplier of a = 1⁄10. Conversely, comfort loss $L^t_{comfort}$ is measured in Watts with a multiplier of b = 1⁄5e5. The penalty variable is defined as c = 1 for **algorithm 1**, whereas **algorithm 2** utilizes c = -50 + (10 **if** A^t_{chp} > 0.5 **else** 0). The framework operates on a 15-min discrete-time control horizon. Furthermore, the state-space S undergoes normalization, and all executable variables in the action-space A are proportionally adjusted to fit within a [+1, -1] range. The scalarization weights applied here align with those established by Ceusters *et al*. [3].

For the learning architecture, we implemented a twin delayed deep deterministic policy gradient (TD3) agent—widely recognized as a cutting-edge RL methodology—sourced from the stable baseline library [40]. The selected hyperparameters for this setup are cataloged in **Table 3**. These values were established following a hyper-parameter optimization process, akin to the procedures in [3], spanning both **Algorithms 1 and 2**.

Crucially, these RL agents do not require an initial “offline” pre-training phase dependent on historical data. Instead, they undergo safe “online” training (executed within a simulated setting). Consequently, the RL agents begin operations devoid of any preexisting environmental insight, dynamically constructing their own training dataset (experience tuples) through active engagement with the environment (represented by the multi-energy system in this study).

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

Table 3. TD3 hyper-parameters.

Hyper-parameters (TD3)	Optsafe	Unsafe	Algorithm 2	Algorithm 1
Gamma	0.7	0.9	0.95	0.7
Learning rate	0.000583	0.0003833	0.000119	0.000583
Batch size	16	100	16	16
Buffer size	1e6	1e5	1e5	1e6
Training frequency	1e0	2e3	1e1	1e0
Gradient steps	1e0	2e3	1e1	1e0
Noise type	normal	normal	normal	normal
Noise standard deviation	0.183	0.329	0.791	0.183

Evaluation

The effectiveness of our proposed techniques, **Algorithms 1 and 2**, is assessed by evaluating their ability to minimize energy costs while adhering to thermal comfort constraints. These are benchmarked against an unrestricted (and potentially hazardous) RL agent, the OptLayer RL agent introduced by Pham *et al*. [34] (noted in related literature), and both safe and unsafe random agents. The random agents function as baseline learning metrics for their respective algorithms (indicating the amount of learning achieved through safe environmental interaction) and represent the baseline performance before any training phase. Consequently, these random agents are not proposed as practical optimal controllers. The experimental setup utilizes a one-year training period, a 15-min control horizon (meaning the energy management RL agent can choose new actions every 15 simulated minutes4), and a one-week evaluation phase within a day-ahead electricity market framework. Inherent uncertainties (such as fluctuations in demand, pricing, or renewable generation) are naturally managed by the RL agent through its formulation as a discrete time-invariant infinite-horizon stochastic optimal control problem (refer to Ref. [3] for the transformation from a continuous time-varying stochastic system). The model-predictive controller discussed in Ref. [3] is excluded from this comparison, as our approach allows for direct constraint formulation within the methodology.

Results and Discussion

The simulation outcomes regarding the objective values (rewards) are presented in **Table 4**, displaying both absolute figures and values relative to the unrestricted RL benchmark. The tolerance for the inequality constraint, **Eq. 3c**, is expressed as a proportion of the total demand (where 0% indicates strict equality constraint fulfillment, meaning all thermal demands are met, including energy routed to thermal storage).

The data in **Table 4** demonstrates that **Algorithm 1** (SafeFallback: 102.9%) surpasses **Algorithm 2** (GiveSafe: 98.2%) and the standard unsafe TD3 baseline (100%), although it slightly trails OptLayer (104.0%). This outcome is logical, given that employing the predefined safe fallback policy yields the greatest utility before any training (94.6% in contrast to 82.8%, 35.5%, and 77.8%), highlighting the value of the embedded expert knowledge. This supplementary expert insight (inherent to the safe fallback policy of **Algorithm 1** itself—see πsafe) also manifests in an initially elevated constraint tolerance (7.0%), which eventually stabilizes at a satisfactory 10.1% (well below the 15% maximum threshold established in **Eq. 3c**). **Algorithm 2** and OptLayer exhibit higher initial constraint tolerances (15.0% and 15.6%, respectively) but ultimately converge to comparable levels (10,0% and 10,4%). Unsurprisingly, the SafeFallback policy (πsafe) inherently

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

maintains the strictest constraint tolerance (6.3%). The elevated tolerance observed in SafeFallback Random (and consequently SafeFallback TD3 before training) is attributable to its capacity to randomly select safe actions, which remain constrained by the 15% upper limit. Thus, as designed, both proposed techniques offer substantially greater safety than the unconstrained vanilla TD3 benchmark, which begins with a massive 146.0% constraint tolerance and only improves to 21.0% (reflecting an unsafe training process devoid of hard-constraint guarantees, an unacceptable condition in real-world scenarios, but useful here strictly as a baseline). It is worth noting that OptLayer initially breaches the 15% maximum limit set in **Eq. 3c**. This violation occurs because OptLayer relies on solving a mixed-integer quadratic problem to determine the closest feasible actions. Furthermore, OptLayer employs linear analytical approximations rather than the surrogate functions f (.) from **Eq. 3c** associated with the metrics in **Table 2,** due to a lack of derivative data.

Table 4. 5-run average policy performance with a training budget of 15 years' worth of time steps per run (i.e. 525.150 time steps per run).

EMS algorithm	Constraint	Objective value	
	Tolerance	Relative	Absolute
Unsafe TD3	21.0%	100.0%	−5.043
Unsafe Random	146.0%	35.5%	−14.223
OptLayer TD3	10.4%	**104.0%**	**−4.850**
OptLayer Random	15.6%	77.8%	−6.481
SafeFallback TD3	10.1%	102.9%	−4.899
SafeFallback Random	7.0%	94.6%	−5.331
SafeFallback (π^{safe})	**6.3%**	96.5%	−5.228
GiveSafe TD3	10.0%	98.2%	−5.137
GiveSafe Random	15.0%	82.8%	−6.089

The training trajectories of the TD3 agents (implementing **Algorithms 1, 2**, OptLayer, and a baseline lacking any safety layer—indicated as UnSafe) are illustrated in **Figure 5.** The starting (time step 0) and concluding (time step 525.150) data points correspond to the values detailed in **Table 4**. We note an initially rapid learning phase, minimal variance, and steady (marginally improving) efficacy as the agent interacts more extensively with the environment. Furthermore, **Algorithm 1** (SafeFallback), **Algorithm 2** (GiveSafe), and OptLayer exhibit substantially superior initial performance (before training at time step 0) when contrasted with the vanilla unsafe TD3 agent. This advantage stems from the integration of preliminary expert knowledge, supplied via the known safe fallback policy and the safety constraint equations. The unsafe RL agent requires approximately 35.000 time steps (1 year) to match OptLayer's starting performance (−6.481). It takes roughly 50.000 time steps (1 year and 5 months) to equal **Algorithm 2**'s starting benchmark (−6.089), and about 85.000 time steps (2 years and 5 months) to reach **Algorithm 1**'s initial level (−5.331). Additionally, OptLayer's baseline performance falls below that of **Algorithms 1 and 2**; it eclipses **Algorithm 2** after roughly 18.575 time steps (6 months) and surpasses **Algorithm 1** after approximately 30.000 time steps (10 months). The performance disparity remains notable against **Algorithm 2** (5.8%), whereas **Algorithm 1** achieves a comparable final performance (1.1%).

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press



Figure 5. 5-run average learning curves with a training budget of 15 years' worth of time steps per run (i.e. 525.150 time steps per run). Note that the y-axis is zoomed in

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

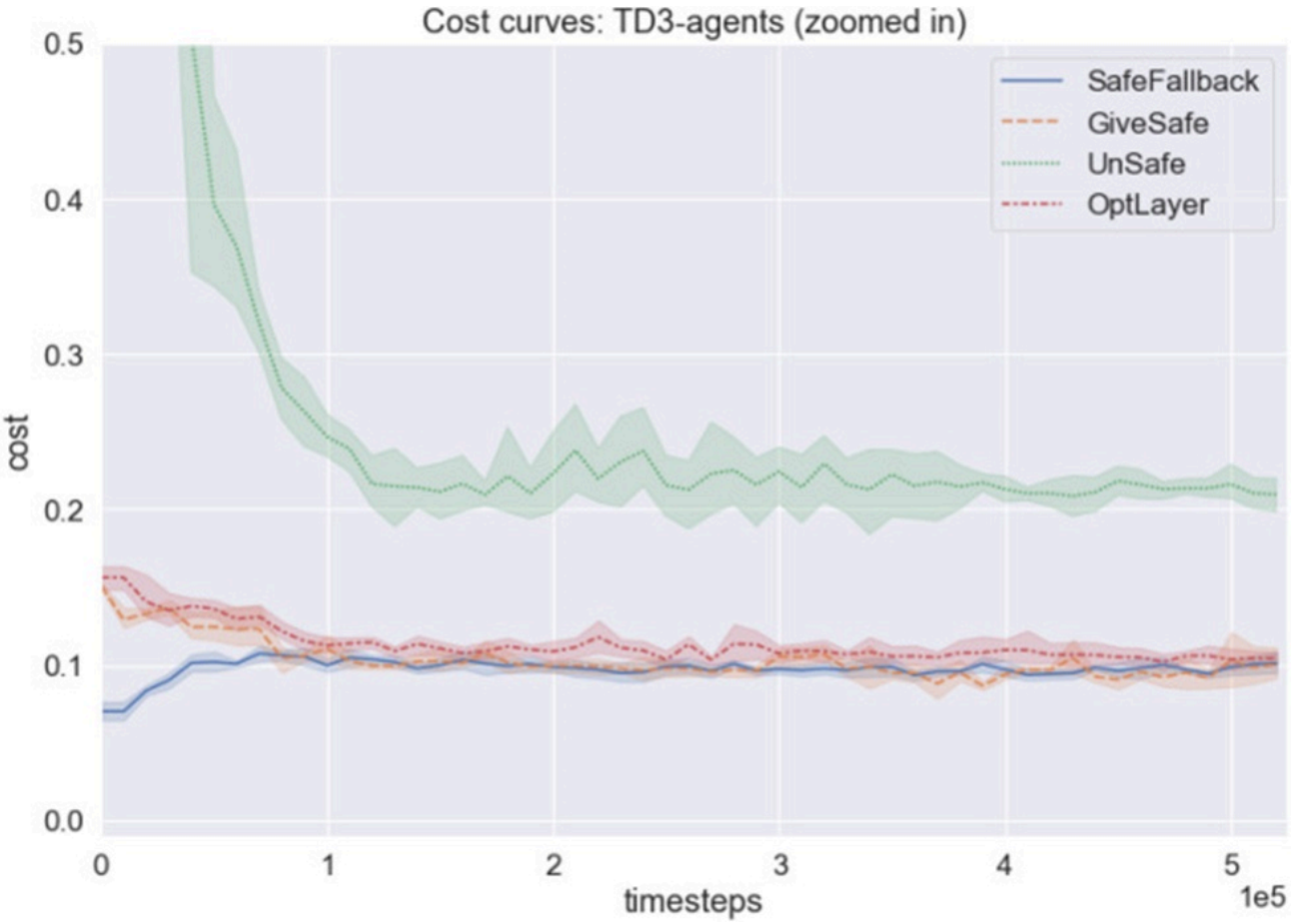


Figure 6. 5-run average cost curve (i.e., constraint tolerance) with a training budget of 15 years' worth of time steps per run (i.e., 525.150 time steps per run). Note that the -axis is zoomed in

Figure 6 illustrates the cost trajectories (constraint tolerances) for the respective TD3 agents. The vanilla TD3 agent demonstrates a sharp initial drop in constraint tolerance but never achieves the 15% safety boundary mandated by **Eq. 3c**. Conversely, **Algorithms 1 and 2** strictly maintain compliance below this threshold (validating their capacity for hard-constraint satisfaction during the training phase). At the same time, OptLayer marginally exceeds it, as previously discussed. The safe methods exhibit a more gradual convergence in constraint tolerance, settling at a consistent level (∼ 10%) following roughly 3 years (∼ $1e5$ time steps). Consequently, if the L_comfort^t variable were removed, the constraint tolerance would drift toward the 15% maximum limit, because a pure cost-minimization strategy would aggressively curtail expensive energy usage (in the absence of constraints or the L_comfort^t factor, all controllable energy assets would shut down during periods of positive energy pricing). This dynamic explains why we see both downward-trending cost curves (initiating above the convergence threshold) and an upward-trending cost curve (initiating below the threshold).

It is crucial to acknowledge, however, that the success of **Algorithm 1** (SafeFallback), **Algorithm 2** (GiveSafe), and OptLayer — in terms of both overall utility (objective value) and cost control (constraint tolerance) — hinges on the precise definition of the actual constraints, specifically the accurate modeling of **Eq. 3c** within this study. Establishing this precision can be complex, particularly regarding the TESS and the HP components. If the pre-trained functions for the TESS and HP within the safety layer are replaced with more rudimentary (linear) analytical equations (identical to those used in OptLayer), performance degrades. For instance, under **Algorithm 1**, the initial objective value shifts from −5.331 to −5.431, and the initial constraint tolerance worsens from 7,0% to 7,6%. This degradation can be partially offset, however, by artificially reducing the Qtol parameter in **Eq. 3c.**

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

Nonetheless, deploying an RL agent for training within a genuine safety-critical setting is only feasible with a highly accurate safety framework (i.e., robust constraints) and appropriate, albeit initially unknown, hyperparameters. Consequently, we outline the following avenues for subsequent research:

· Establishing chance-constraint satisfaction guarantees for scenarios where predefined constraint functions are unavailable (relying instead on restricted, verified-safe operational data) and methodically refining these initial constraint models as additional data is gathered. Under this framework, a statistical model (such as a Gaussian Process model) would be continuously updated to ensure that exploratory and exploitative actions remain within the safe operational envelope with a high degree of probability.

· Minimizing the required training budget (thereby boosting sample efficiency, potentially by integrating SafeFallback with OptLayer) and implementing a predetermined series of (robust5) control actions (for instance, through model-based RL agents) to facilitate day-ahead market scheduling.

· Enhancing the resilience of RL-driven energy management frameworks against erroneous or noisy data inputs and incorporating online hyper-parameter optimization techniques (allowing hyperparameters to adjust dynamically throughout the online training phase).

Conclusion

This research introduced two innovative model-free safe reinforcement learning (RL) strategies characterized by the following features: (i) they ensure hard-constraint satisfaction across multiple constraints, rather than relying on soft or chance constraints; (ii) they operate independently from the primary RL (Markov Decision Process) architecture; (iii) they maintain safety both throughout the training of a (near) optimal policy (accommodating exploratory and greedy exploitative actions) and during the execution of any given policy (including random agents or pre-trained offline RL agents); and (iv) they achieve this without necessitating the resolution of a mathematical program. The efficacy of these techniques was showcased within the context of a multi-energy management system, supported by comprehensive simulation data.

Both proposed approaches represent practical constraint-handling mechanisms that expand upon current state-of-the-art RL methodologies, successfully providing rigid safety guarantees even when applied to random agents. However, **Algorithm 1** (SafeFallback) emerges as the preferred option. It demonstrates strong performance, bypasses the need for solving complex mathematical programs (unlike OptLayer, which requires a mixed-integer quadratic program), and capitalizes on the fact that rudimentary safe fallback policies are typically either already available or straightforward to engineer (often manifesting as a basic rule-based framework, such as a priority-driven control protocol).

Acknowledgements

This work has been supported in part by ABB n.v., Belgium and Flemish Agency for Innovation and Entrepreneurship (VLAIO) grant HBC.2019.2613, Belgium.

Conflict of interest

None

Financial support

None

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

Ethics statement

None

Received: 18 Feb 2024 Revised: 25 Apr 2024 Accepted: 10 May 2024
Published online: 15 November 2025

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at https://creativecommons.org/licenses/by-nc-sa/4.0/.

References

1. Fabrizio E, Filippi M, Virgone J. Trade-off between environmental and economic objectives in the optimization of multi-energy systems. Build Simul. 2009;2(1):29-40.

2. Engell S. Feedback control for optimal process operation. J Process Control. 2007;17(3):203-19.

3. Ceusters G, Rodríguez RC, García AB, Franke R, Deconinck G, Helsen L, et al. Model-predictive control and reinforcement learning in multi-energy system case studies. Appl Energy. 2021;303:117634.

4. Cao D, Hu W, Zhao J, Zhang G, Zhang B, Liu Z, et al. Reinforcement learning and its applications in modern power and energy systems: A review. J Mod Power Syst Clean Energy. 2020;8(6):1029-42.

5. Yang T, Zhao L, Li W, Zomaya AY. Reinforcement learning in sustainable energy and electric systems: a survey. Annu Rev Control. 2020;49:145-63.

6. Perera AT, Kamalaruban P. Applications of reinforcement learning in energy systems. Renew Sustain Energy Rev. 2021;137:110618.

7. Zhou Y. Advances of machine learning in multi-energy district communities–mechanisms, applications and perspectives. Energy AI. 2022;10:100187.

8. Rayati M, Sheikhi A, Ranjbar AM. Applying reinforcement learning method to optimize an energy hub operation in the smart grid. In: 2015 IEEE Power and Energy Society Innovative Smart Grid Technologies Conference (ISGT). 2015.

9. Watkins CJCH. Learning from delayed rewards [PhD thesis]. Cambridge (UK): King's College; 1989.

10. Sheikhi A, Rayati M, Ranjbar AM. Demand side management for a residential customer in multi-energy systems. Sustainable Cities Soc. 2016;22:63-77.

11. Mbuwir BV, Kaffash M, Deconinck G. Battery scheduling in a residential multi-carrier energy system using reinforcement learning. In: 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm). Institute of Electrical and Electronics Engineers Inc.; 2018.

12. Wang X, Chen H, Wu J, DIng Y, Lou Q, Liu S. Bi-level multi-agents interactive decision-making model in regional integrated energy system. In: 2019 3rd IEEE Conference on Energy Internet and Energy System Integration: Ubiquitous Energy Network

Connecting Everything (EI2). Institute of Electrical and Electronics Engineers Inc.; 2019. p. 2103-8.

13. Ahrarinouri M, Rastegar M, Seifi AR. Multi-agent reinforcement learning for energy management in residential buildings. IEEE Trans Ind Inf. 2020;1.

14. Ye Y, Ye Y, Qiu D, Wu X, Strbac G, Ward J. Model-free real-time autonomous control for a residential multi-energy system using deep reinforcement learning. IEEE Trans Smart Grid. 2020;11(4):3068-82.

15. Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. In: 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings. International Conference on Learning Representations, ICLR; 2015.

16. Xu Z, Han G, Liu L, Martinez-Garcia M, Wang Z. Multi-energy scheduling of an industrial integrated energy system by reinforcement learning-based differential evolution. IEEE Trans Green Commun Netw. 2021;5(3):1077-90.

17. Zhu D, Yang B, Liu Y, Wang Z, Ma K, Guan X. Energy management based on multi-agent deep reinforcement learning for a multi-energy industrial park. Appl Energy. 2022;311:118636.

18. Pu Y, Wang S, Yang R, Yao X, Li B. Decomposed soft actor-critic method for cooperative multi-agent reinforcement learning. arXiv. 2021.

19. Zhou Y, Ma Z, Zhang J, Zou S. Data-driven stochastic energy management of multi energy system using deep reinforcement learning. Energy. 2022;261:125187.

20. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. arXiv. 2017.

21. Fujimoto S, Van Hoof H, Meger D. Addressing function approximation error in actor-critic methods. In: 35th International Conference on Machine Learning. Vol. 4. International Machine Learning Society (IMLS); 2018. p. 2587-601.

22. Zhang B, Hu W, Cao D, Li T, Zhang Z, Chen Z, et al. Soft actor-critic –based multi-objective optimized energy conversion and management strategy for integrated energy systems with renewable energy. Energy Convers Manage. 2021;243:114381.

23. Zhang B, Hu W, Cao D, Huang Q, Chen Z, Blaabjerg F. Deep reinforcement learning–based approach for optimizing energy conversion in integrated electrical and heating system with renewable energy. Energy Convers Manage. 2019;202:112199.

24. Zhang B, Hu W, Li J, Cao D, Huang R, Huang Q, et al. Dynamic energy conversion and management strategy for an integrated electricity and natural gas system with renewable energy: Deep reinforcement learning approach. Energy Convers Manage. 2020;220:113063.

25. Zhang G, Hu W, Cao D, Zhang Z, Huang Q, Chen Z, et al. A multi-agent deep reinforcement learning approach enabled distributed energy management schedule for the coordinate control of multi-energy hub with gas, electricity, and freshwater. Energy Convers Manage. 2022;255:115340.

26. García J, Fernández F. A comprehensive survey on safe reinforcement learning. J Mach Learn Res. 2015;16:1437-80.

27. Dulac-Arnold G, Levine N, Mankowitz DJ, Li J, Paduraru C, Gowal S, et al. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. Mach Learn. 2021;110(9):2419-68.

28. Nweye K, Liu B, Stone P, Nagy Z. Real-world challenges for multi-agent reinforcement learning in grid-interactive buildings. Energy AI. 2022;10:100202.

29. Brunke L, Greeff M, Hall AW, Yuan Z, Zhou S, Panerati J, et al. Safe learning in robotics: From learning-based control to safe reinforcement learning. Ann Rev Control Robot Auton Syst. 2021;5(1).

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press

30. Venayagamoorthy GK, Sharma RK, Gautam PK, Ahmadi A. Dynamic energy management system for a smart microgrid. IEEE Trans Neural Netw Learn Syst. 2016;27(8):1643-56.

31. Zhang Q, Dehghanpour K, Wang Z, Huang Q. A learning-based power management method for networked microgrids under incomplete information. IEEE Trans Smart Grid. 2020;11(2):1193-204.

32. Zhao H, Zhao J, Qiu J, Liang G, Dong ZY. Cooperative wind farm control with deep reinforcement learning and knowledge-assisted learning. IEEE Trans Ind Inf. 2020;16(11):6912-21.

33. Park H, Min D, Ryu Jh, Choi DG. DIP-QL: A novel reinforcement learning method for constrained industrial systems. IEEE Trans Ind Inf. 2022.

34. Pham TH, De Magistris G, Tachibana R. OptLayer - practical constrained optimization for deep reinforcement learning in the real world. In: Proceedings - IEEE International Conference on Robotics and Automation. Institute of Electrical and Electronics Engineers Inc.; 2018. p. 6236-43.

35. Mattsson SE, Elmqvist H, Otter M. Physical system modeling with Modelica. Control Eng Pract. 1998;6:501-10.

36. Gräber M, Fritzsche J, Tegethoff W. From system model to optimal control - A tool chain for the efficient solution of optimal control problems. In: Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017. Vol. 132. Linköping University Electronic Press; 2017. p. 249-54.

37. Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, et al. OpenAI Gym. arXiv. 2016.

38. Lukianykhin O, Bogodorova T. ModelicaGym: Applying reinforcement learning to Modelica models. In: ACM International Conference Proceeding Series. Association for Computing Machinery; 2019. p. 27-36.

39. Andersson C, Akesson J, Fuhrer C. PyFMI: A python package for simulation of coupled dynamic models with the functional mock-up interface: Technical Report in Mathematical Sciences. Lund University; 2016. p. 1-40.

40. Raffin A, Hill A, Gleave A, Kanervisto A, Ernestus M, Dormann N. Stable-Baselines3: Reliable reinforcement learning implementations. J Mach Learn Res. 2021;22(268):1-8.

© 2025 The Author(s). | Published by Institute of Smart Energy Systems Press