15TH IEA HEAT PUMP CONFERENCE
MAY 26–29, 2026 | HOFBURG VIENNA 2026
DECARBONISATION THROUGH INNOVATION

Evaluating heat pump-enabled building flexibility potential via deep reinforcement learning and trade-off analysis

Lorenz Ray Payonga[a,*], Hatef Madani[a]

[a]KTH Royal Institute of Technology, Department of Energy Technology, Stockholm

Abstract

Electricity grids increasingly face volatility due to supply-demand imbalances and congestion. To manage this, grid operators seek flexibility not only from power plants and batteries but also from underutilized resources, such as the thermal inertia of residential buildings equipped with heat pumps. This study explores how these systems can contribute to grid stability while balancing consumer objectives like comfort, cost savings, and environmental impact reduction. We propose a Deep Reinforcement Learning (DRL) approach to model and optimize heating control strategies. An agent learns to determine optimal heating actions and temperature setpoints based on different reward structures. These include comfort-oriented control (maintaining indoor temperature), cost minimization, and reduction of environmental impact. We extend this to include flexibility maximization, where the agent adapts its behavior to grid flexibility requirements. The approach is validated through a simulation environment. Results demonstrate the agent's ability to analyze and negotiate trade-offs between competing objectives and respond dynamically to grid flexibility requirements. This work serves as a demonstration of embedding trade-off analysis in evaluating and activating flexibility.

Selection and/or peer-review under the responsibility of the organizers of the 15th IEA Heat Pump Conference 2026.

Keywords: flexibility, reinforcement learning, trade-offs, optimization, adaptive control

1. Introduction

In view of the volatility in the electricity grid due to supply-demand imbalances and congestion caused by an evolving energy system, flexibility services have been positioned as a viable alternative to grid reinforcement [1]. The provision of such services is often discussed on a large scale, like contractually-provided balancing services. It is, however, well-established that individual buildings particularly in the residential sector also have the potential to activate flexibility, especially those whose heating systems are electrically-coupled through heat pumps [2]. With the introduction of the aggregator role in the EU Internal Electricity Market Directive of 2019, this potential can be turned into actual flexibility services, especially in short and medium-term timescales [3].

Quick flexibility response is a matter of importance to distribution system operators (DSOs) considering that network needs can appear or increase rapidly at any given location and that it is critical to have access to needed resources in a timely manner at specific locations [4]. In this context, local flexibility services are invaluable, which opens an opportunity for residential buildings to contribute. At this timescale, intraday markets for flexibility will be essential. Although there are a few attempts to evaluate the potential of residential buildings with heat pumps for offering flexibility through the intraday market [5], most of the known applications were in the day-ahead market [2].

Flexibility activation is often framed as customers' contribution to keeping the grid stable. While it is a noble act, priorities of and impacts on those that provide such a service cannot be overlooked. Particularly in

* Corresponding author. Tel.: +46-764-528-801
E-mail address: payonga@kth.se

the residential sector, a deviation in energy use will inevitably have corresponding consequences on comfort, energy costs, and environmental impact – three of the most salient considerations in household energy efficiency [6]. Related studies include control and modelling approaches to optimizing for one of these objectives [7], post-facto evaluations of temperature deviations due to heat pump-activated flexibility [8], and an estimation of compensation costs for such deviations [9]. To our knowledge, there has been no study so far that embeds balancing these objectives into a control strategy specifically for flexibility activation.

One suitable approach of developing control models for the abovementioned gap is through *reinforcement learning* (RL). This machine learning (ML) method is built on *trade-off exploration/exploitation*, where an *agent* initially explores random actions and then eventually "learns" a *policy* that exploits actions that are highly "rewarded" in training [10]. In Deep RL, instead of hard-coding rules or using historical data, a *neural network*'s parameters are tuned by numerical rewards that are designed according to the potentially conflicting objectives of comfort, costs, and environmental impact. Deep RL's applications in home energy management and heat pump control [11, 12] and energy storage [12] have been explored, but the specific application for flexibility activation remains limited.

Therefore, in this paper, we addressed the following objectives:

- Train through Deep RL a high-level heat pump and space heating controller that can respond to flexibility requests while still pursuing user-centric optimization objectives;
- Analyze the trade-offs between optimization modes; and
- Evaluate the user-side trade-offs when providing flexibility.

It is worth mentioning that the results will not focus on learning statistics typical of RL papers, but on trade-off analysis among the resulting models. This work also does not account for any incentives that may be offered by the DSO in exchange for flexibility. Nonetheless, it is aimed for this work to offer DSOs and other flexibility requesting parties (FRPs) quantitative perspective that can help inform incentive design and a more comprehensive cost-benefit analysis that recognizes user-side trade-offs.

The paper is structured as follows: Section 2 outlines the different component methods including system modelling, RL implementation, model selection, and flexibility and trade-off analysis. Section 3 discusses the results of the trained flexibility models and compares their outcomes and computed trade-offs. Evidence of adaptive control is also provided. We then conclude and recommend future directions.

2. Methods/Approach

Training an RL agent requires an *environment* that provides new *states* and *rewards* in response to the agent's *actions*. The agent selects actions based on the states it observes, typically using a *policy* represented by a *neural network*. The rewards returned by the environment are used by the RL *algorithm* to update the neural network's weights and biases so that the policy improves over time. Thus, it is necessary to structure the training process such that the RL agent receives realistic state observations for the actions it performs, which often starts with *exploration* (stochastic) until the policy converges (i.e., *exploitation*).

In the following sections, the elements of the training process are described, including the co-simulation framework, model selection and trade-offs analysis, and flexibility evaluation.

2.1. RL training and co-simulation framework

The RL training and co-simulation framework is described in Fig. 1, which is partly based on the framework used by Bousnina & Guerassimoff [13]. A system model was developed in Dymola using the Modelica language, which was exported as a functional mock-up unit (FMU). This allows the simulation of system dynamics to be performed independently of the original modeling tool and allows straightforward integration with the training environment. Component actions related to system components and state observations are obtained and relayed, respectively, by the custom environment to and from the FMU through the FMPy API [14]. The system model is described in Section 2.1.1.

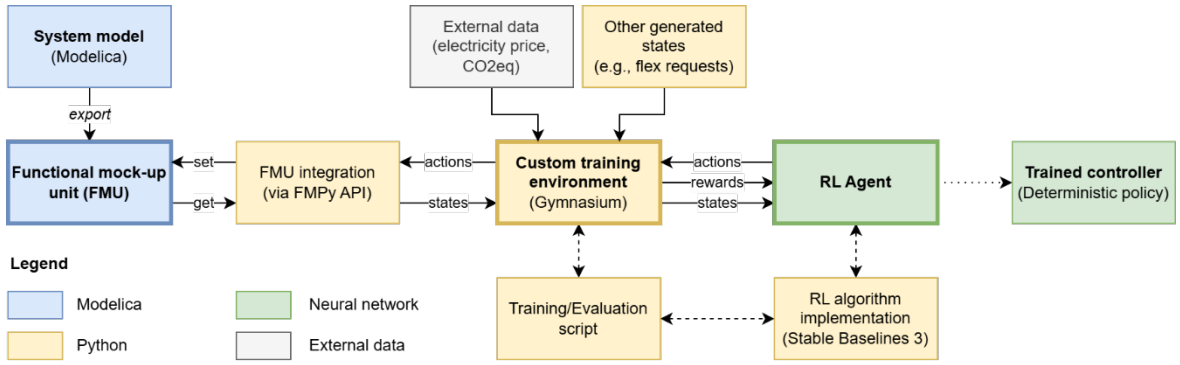

Figure 1. RL training and co-simulation framework.

Apart from the FMU, other state observations such as electricity price, CO2 emissions equivalent, and flexibility requests are loaded into and iterated by the custom training environment. Reward functions are defined in the environment. The RL agent was trained through a script which uses the *Proximal Policy Optimization (PPO)* algorithm as implemented through the Stable Baselines 3 framework [15]. Similar to [16], an in-training evaluation approach was implemented, with key performance indicators (KPIs) that served as criteria for selecting a model (i.e., neural network parameters) to be saved and later run as a deterministic policy in evaluation and deployment. The training environment, algorithm specifications, and model selection are described in sections 2.1.3, 2.1.4, and 2.2, respectively.

External data that were used include the following: 2024 hourly electricity spot prices from Nordpool, typical meteorological year (TMY) weather file for Stockholm (loaded into the system model) [17], and grid carbon intensity data from 2023 [18].

2.1.1. System model

The system model used to dynamically generate state observations contains a load based on a Swedish single-family house building archetype from the TABULA Webtool [19] and a heating system based on equipment available at the KTH Department of Energy Technology. In the Modelica model, the load is described with lumped parameters, i.e., room volume of 265.5 m^3 (106.2 m^2 conditioned floor area), 17,204.4 kJ/K (45 Wh/m^2K) heat capacity, 113W/K transmission heat transfer coefficient (0.238 W/m^2K weighted U-value), and solar heat gain calculated based on provided reduction factors, transmittance, and area per orientation. The radiator size of 7 kW was determined from observed heat demand in initial model simulations.

The heat is supplied by a 12-kW water-to-water heat pump model with a scroll compressor from the Modelica Buildings Library [20], which approximates the 12-kW heat pump available at the KTH laboratory. With an evaporator source temperature fixed at 10°C and a 2 K temperature drop across the evaporator on the source side (10°C to 8°C), the heat pump supplies heat to a 0.75 m^3 hot water buffer tank.

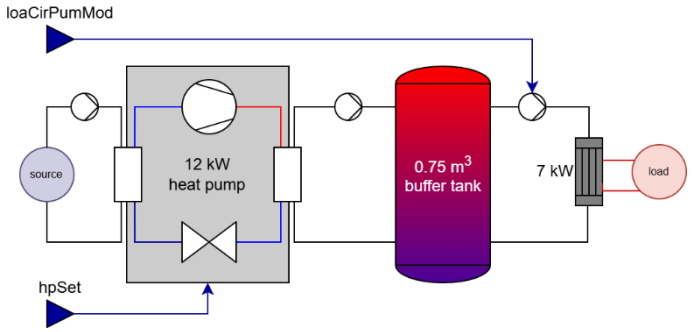

Figure 2. Simplified heat pump-driven building space heating system with buffer tank, showing the two actions decided by the RL agent.

As can be seen in Fig. 2, two parameters are exposed through interfaces: the heat pump supply temperature setpoint (*hpSet*) and the load-side circulation pump modulation setpoint (*loaCirPumMod*). These interfaces take in *set* commands through the FMPy API based on the custom environment's interpretation of the agent's actions. To mimic the specifications of the heat pump at the KTH laboratory, *hpSet* corresponds to the target supply temperature based on a predetermined heating curve centered at 40°C supply at 0°C outdoor temperatures, which can be offset by a user (or an external controller) such that a unit of change is equivalent to +/- 3K offset of the heating curve. Meanwhile, *loaCirPumMod* goes from 0.0 to 1.0 in 0.1 increments.

2.1.2. Flexibility concepts

Before getting into the details of the training environment, it is necessary to explain what "flexibility" means in the context of this work. Unless otherwise clarified, flexibility and its related terms "flexibility potential" and "flexibility activation" refer to the deviation from a forecasted demand profile a customer (e.g., "active customer", aggregator) can deliberately provide within the bounds of flexibility allocated by an FRP, which in the case of distribution grid congestion is a DSO. The way by which flexibility is requested is described in the USEF Flexibility Trading Protocol (UFTP) [21].

In the UFTP, an aggregator offers flexibility by first providing a status quo forecast of the congestion point to the DSO. This is called the "*D-prognosis*". The DSO processes this forecast, evaluates possible congestion in terms of imbalance settlement periods (ISP) (or program time units, PTU, in electricity market terms), and issues a flexibility request which contains the ISP, and the minimum and maximum allowed power demand at those times.

2.1.3. Training environment

2.1.3.1. State space

The custom training environment *BuildingEnv* has a state space with variables that can be categorized into time observations (4), thermal measurements (4), electrical measurements (3), price/cost data (2), grid emissions data (2), weather data (2), flexibility request parameters (4), device status (2), and settings and benchmarks (5). These can be further classified into either endogenous or exogenous types, depending on whether they are directly affected by the agent's actions or not, respectively. These are provided in Table 1.

The state variables were deliberately selected to sufficiently inform the learning process without overspecification. Although more thermal, electrical, and weather measurements can be obtained from the FMU, only a few relevant ones to the agent training objectives were selected. For instance, among weather data, we only used outdoor ambient temperature and solar irradiance because they are known to be the most significant contributors to predicting building heat demand [22]. Among available thermal energy measurements, the energy supplied at the load (*load_sup_ener_dt*) was used to represent an immediate change in state due to a circulation pump action, despite the expected delay in indoor or tank temperatures due to thermal inertia [23].

Table 1. State space of BuildingEnv

Category	State variables	Type
Time	day_sin, day_cos, year_sin, year_cos	Exogenous
Thermal measurements	temp_tank, temp_indoor, temp_indoor_dt*, load_sup_ener_dt	Endogenous
Electrical measurements	elec_demand, elec_demand_h*, elcons_change*	Endogenous
Price data	elec_price, price_change*	Exogenous
Grid emissions data	co2_val, co2_change*	Exogenous
Weather data	temp_outdoor, irradiance_ghi	Exogenous
Flexibility request parameters	flex_req_signal, flex_limit_min, flex_limit_max, forecast_diff*	Exogenous, except forecast_diff
Device status	load_circ, temp_set_tank	Endogenous
Settings and benchmarks	mode, temp_set, elec_demand_nom, base_price, base_co2	Exogenous

*augmented states

Additionally, it can be noticed from the table above that time was cyclically encoded to ensure temporal continuity; thus, having sine and cosine components for a day to account for diurnal variations and the year to capture seasonal variations. Having properly-specified temporal variables also ensures that other states' time dependency is also accounted for. Meanwhile, variables representing change, i.e., *temp_indoor_dt*, *elcons_change*, *price_change*, *co2_change*, and *forecast_diff*, and the variable *elec_demand_h* representing the cumulative electrical energy consumption from the start of every hour were computed within *BuildingEnv* and were generated as separate state observations to address partial observability in the context of thermal inertia and the objectives of optimizing for cost, environmental impact, and flexibility response. The technique of "state augmentation" has been shown to increase robustness in RL without violating the Markov principle [24].

As regards flexibility parameters *flex_req_signal*, *flex_limit_min*, and *flex_limit_max* are externally processed variables that are loaded into *BuildingEnv,* where flexibility requests are generated based on a congestion schedule that follows the UFTP message structure. This external processing by a "flexibility request generator" described in [25] takes in the *D-prognosis* for a duration of expected congestion as well as the minimum and maximum power demand allocated for a particular congestion point at specific PTUs (as explained in Section 2.1.2). Times of congestion are labelled as *flex_req_signal* = 1, while *flex_limit_min* and *flex_limit_max* are computed allowable deviations from the forecasted electricity demand. During training, the augmented state *forecast_diff* is the computed difference between forecasted cumulative electrical energy consumption (also loaded into *BuildingEnv*) for the PTU and that of the actual (i.e., *elec_demand_h*).

The relevance of the "device status" state variables is elaborated with the action space description (Section 2.1.3.2), while the use of "settings and benchmarks" and the inclusion of the augmented states are further motivated in the section on reward functions (Section 2.1.3.3).

2.1.3.2. Action space

The agent outputs a 2-dimensional discrete action, consisting of the 1) offset action and the 2) load circulation pump modulation signal. The offset action takes an integer value of 0 to 6, which is internally mapped by *BuildingEnv* into the offset value (i.e., -3 to 3) and converted into the heat pump supply temperature setpoint according to the heating curve. The circulation pump action takes an integer value from 0 to 10, which is normalized to get the appropriate modulation signal sent to the circulation pump. The direct outcomes of these actions are returned as the device status state variables *temp_set_tank* and *load_circ*, which are equivalent to the current states of *hpSet* and *loaCirPumMod* in the system model.

2.1.3.3. Reward functions and principles

Table 2. Descriptions of reward functions

Mode	Description
Comfort	Incentivize less deviation of *temp_in* from *temp_set* (21°C)
(*mode* = 1)	Penalize when *temp_in* is outside of acceptable range
	Incentivize or penalize when *load_circ* operates in line or contrary to heating demands based on *temp_in*
	In modes 2 and 3, penalize when *load_circ* is distributing heat when the *temp_tank* < *hpSet*
Economy	Incentivize when *elcons_change* and *price_change* are in opposite directions; penalize otherwise
(*mode* = 2)	Incentivize when *elec_demand* < benchmark electricity demand (as a proportion of *elec_demand_nom*) when *elec_price* > *base_price* (0.40SEK/kWh) or vice-versa; penalize otherwise
Environment	Incentivize when *elcons_change* and *co2_change* are in opposite directions; penalize otherwise
(*mode* = 3)	Incentivize when *elec_demand* < benchmark electricity demand (as a proportion of *elec_demand_nom*) when *co2_val* > *base_co2* (25gCO2/kWh) or vice-versa; penalize otherwise
Flexibility	Incentivize when *forecast_diff* is within *flex_limit_min* and *flex_limit_max* during *flex_req_signal*
(Stage 3)	Incentivize non-deviation from the forecast (i.e., *forecast_diff* = 0) when no *flex_req_signal*

The general structure of the reward function is:

$$r_{total} = r_{comfort} + r_{economy/environment} + r_{flexibility}$$

The reward functions differ depending on the *mode* and the stage of training (elaborated in Section 2.1.4). Instead of going over the individual reward functions, descriptive explanations are provided in Table 2. Reward shaping principles were observed in coming up with the reward functions.

2.1.4. Training approach and algorithm

A curriculum learning (CL) approach was used to train the RL agent, in which the model is trained in three progressively more complex stages. This strategy, also applied in [25], primarily addressed the problem of *sparse rewards* associated with intermittent flexibility requests by enabling the agent to first master simpler sub-tasks before tackling the full multi-objective control problem. As shown in Fig. 3, the agent first learns to maintain thermal comfort (Stage 1). Using the same model, additional reward components related to economic or environmental objectives are then introduced (Stage 2), allowing the agent to extend its behavior while retaining the previously learned comfort-maintenance policy. Finally, the agent is exposed to dynamic flexibility parameters based on randomly-generated congestion schedules that change in every simulation run (Stage 3). In addition, the training weather data spans progressively colder conditions, from September to December, which provides a balanced range of experiences: initially preventing overheating and gradually requiring more heating actions. This stepwise increase in complexity helps the agent navigate potentially conflicting objectives, thus providing structured exposure to trajectories that balance comfort, cost, environmental performance, and flexibility. This gradual progression reduces the difficulty of exploration and facilitates more stable learning of a policy that can optimize the aggregation of competing reward components.

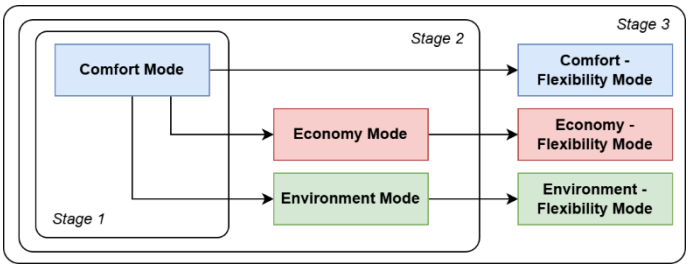

Figure 3. Three-stage curriculum learning approach.

After training and evaluation in each stage, the neural network parameters were saved and could be used for the purpose for which they are optimized. Specifically, there are six files that could be used, corresponding to the modes shown in Fig. 3. However, if the desired controller is one that is responsive to flexibility requests, only those trained at Stage 3 are applicable.

As earlier mentioned, PPO was the RL algorithm used due to its known stability in the training process and its ability to manage noise and outliers. An unpacked implementation of Stable Baselines 3 (SB3) framework was used to run the algorithm, with some modifications in the default hyperparameters such as the number of rollout steps, minibatch size, and number of epochs, which resulted in more stable convergence and desirable KPIs. The PPO algorithm was initially described in [26] and the SB3 implementation is documented in [15]. Except for the unchanged hyperparameters in SB3 PPO, training parameters are provided in Table 3.

Table 3. Training parameters

Parameter	Value	Parameter	Value
FMU communication step size	60 s (1 min)	Environment step size	300 s (5 minutes)
Simulations (Stage 1/2/3)	100/50/75	Rollout length	1440 steps (5 days)
Training weather data	Sep 1 to Dec 30	Minibatch size	48 steps (4 hours)
Eval/Depl. weather data	Jan 1 to Mar 31	Epochs	15

The modifications from default hyperparameters, such as the rollout length, minibatch size, and epochs came about as the most suitable settings for training the control agent, especially in the comfort mode. Specifically, a rollout length of 5 days is sufficient to capture diurnal temperature variations while remaining short enough to preserve sensitivity to short-term deviations rather than overly favoring long-term rewards.

Another thing to note is that although the number of simulations is indicated above, the model which receives the most preferable KPIs would be chosen for deployment.

In-training evaluation would be performed after every co-simulation run beyond the fifth if the agent accumulates rewards above a minimal positive threshold for three of the four simulation months. This is an arbitrary proxy value to ascertain that the system model will not run into errors due to freezing in evaluation while the agent is still largely exploring. Model selection while in evaluation is elaborated in the next subsection.

2.2. Model selection

In the course of in-training evaluation, KPIs collected are as follows:

- Comfort-related: indoor temperature average setpoint deviation (K)*, 90% central range (K)[1]
- Cost-related: total electricity cost (SEK)*, effective electricity rate (SEK/kWh)
- Environment-related: total electricity use (kWh), equivalent CO2 emissions from the grid (kgCO2e), and effective CO2e emissions from the grid (gCO2e/kWh)*
- Flexibility-related: flexibility request response rate, total flexibility activated (kWh)*

For each of Stage 1 and Stage 2 modes, the KPIs related to the respective optimization objectives are used to evaluate whether a model should be saved or not. A model with better performance in the primary KPI (marked with *) overwrites a previously selected one as long as indoor temperature remains in a predetermined tolerance range and the secondary KPIs are not too far from the best-so-far value. In Stage 3, the primary KPI becomes the total flexibility activated (sum of both upward and downward flexibilities) regardless of the original optimization objective. Forecast deviations that are outside of *flex_limit_min* and *flex_limit_max* or those during non-congested PTUs are not counted as activated flexibility. In this case, a fixed randomly-generated congestion schedule is used to evaluate model performance. The saved models act as controller options in deployment.

2.3. Flexibility evaluation and trade-off analysis

For the purposes of demonstrating the performance of the saved models in evaluating and activating flexibility, the simulated building is subjected to a fixed daily 7-hour congestion schedule from 15:00 to 22:00 that allocates an arbitrary range of 0 to 500 Wh/h demand. For simplicity, only the heat pump's electricity consumption is used to represent the building. The models trained until Stage 3 are used to generate the *D-prognosis* assuming no congestion. The same models are then exposed to the flexibility requests generated based on the congestion schedule (as described in Section 2.1.3.1). Apart from the flexibility KPIs, deviations from primary optimization KPIs for each mode, as well as the presence of adaptive control were investigated. This is important because it would validate if the controller driven by the saved models can provide flexibility without prior knowledge of upcoming flexibility requests.

2.4. Benchmark rules-based control

To have a benchmark for comparison, a simple hysteresis-based rules-based control (RBC) was also implemented. The RBC controller switches on the load circulation pump to discharge the hot water buffer tank when the indoor temperature reaches 20°C or lower, while it turns it off when indoor temperature is at least 22°C to prevent further heat discharge. This hysteresis range centered at 21°C aligns with the indoor temperature setpoint for the RL-trained controllers. A -1K heat curve offset was chosen because, among the possible offsets (i.e., -3 to +3K), it has the smallest average setpoint deviation with heating actions conservative enough to keep energy usage lower than at higher offsets.

[1] 90% central range refers to the interval between the 5th and 95th percentiles of the observed values. This metric was used as a robust measure of spread, as it captures 90% of the distribution while reducing sensitivity to extreme values, in contrast to the standard deviation.

3. Results and Discussion

Table 4 summarizes the KPIs for all models trained, including the benchmark RBC controller.

Table 4. Summary results for optimization modes

KPI	RBC	COMFORT		ECONOMY		ENVIRONMENT	
		Comfort	Comfort-Flex	Economy	Economy-Flex	Environment	Environment-Flex
Average setpoint deviation (K)	0.09	0.23	0.50	0.53	0.00	0.49	0.58
90% central range (K)	2.64	1.90	2.82	6.91	6.67	3.18	2.60
Total electricity cost (SEK)	653.22	620.30	622.51	458.45	488.50	511.15	552.16
Effective electricity rate (SEK/kWh)	0.66	0.65	0.63	0.57	0.58	0.61	0.63
Total electricity use (kWh)	989.45	961.67	981.59	801.00	849.19	836.69	872.99
Equivalent CO2 emissions (kgCO2eq)	27.14	26.26	26.79	21.69	23.00	22.65	23.68
Effective CO2e emissions (gCO2eq/kWh)	27.43	27.31	27.29	27.07	27.09	27.07	27.12
Flexibility response rate	-	-	0.63	-	0.15	-	0.63
Flexibility activated (kWh)	-	-	127.98	-	27.88	-	49.70

Table 4 can be understood in three ways, i.e., 1) comparison of RL-trained controllers vs RBC, 2) comparison among optimization modes, and 3) comparison between Stage 2 and Stage 3 modes. Firstly, we compare the Stage 2 control modes (comfort, economy, environment) against the RBC controller. We note that the RBC yielded the lowest average setpoint deviation, which is expected given its predictable hysteresis control centered on the setpoint. However, the RL-trained control modes generally outperformed the RBC controller in overall system performance. All Stage 2 control modes used less electricity and achieved lower total electricity cost, effective electricity rates, and CO_2-equivalent emissions than the RBC controller. Additionally, although the RBC had the smallest average setpoint deviation, the comfort mode produced a tighter temperature distribution while also achieving more favorable KPIs across the board. These results illustrate the state-aware behavior of the RL-trained controller, which pursues the defined optimization objectives while managing trade-offs, in contrast to the fixed rules of the RBC controller.

Secondly, we compare among optimization modes. All control modes behaved consistently with their optimization objectives. For both Stage 2 and Stage 3, the comfort modes prioritized minimizing setpoint deviation and maintaining tight indoor temperature distributions, while the economy modes targeted the lowest electricity costs (while also significantly reducing the CO2-equivalent emissions). The performance of the environment modes requires additional context. Although these modes did not produce the lowest CO_2-equivalent emissions, they still achieved substantially reduced emissions comparable to the economy modes, while maintaining better comfort conditions, indicating a more balanced trade-off between emissions reduction and thermal comfort.

Lastly, we discuss what happens in optimization modes when also pursuing flexibility objectives, i.e., Stage 2 vs Stage 3. In each mode, providing flexibility introduces trade-offs across several KPIs, while the primary objective of each mode generally remains within a reasonable range. Specifically, comfort mode continues to maintain low average setpoint deviation and a tight indoor temperature distribution, economy mode maintains low electricity costs, and environment mode keeps CO_2-equivalent emissions relatively low. Regarding flexibility activation, *economy-flex* provides the least flexibility. This may be because comfort has already degraded to a level where further flexibility would be detrimental, because the model attempted to provide flexibility but the resulting outcomes fell outside acceptable flexibility bounds, or due to a combination of both factors. In either case, this highlights an opportunity to improve the training process and model selection.

Fig. 4 shows how flexibility activation affects indoor temperature, electricity costs, and CO_2-equivalent emissions. Flexibility was quantified by aggregating deviations from the *D-prognosis* in daily electricity consumption (kWh/day) and relating these to the corresponding deviations in the evaluated KPIs as predicted.

The regression trendlines therefore indicate how much each KPI changes for every additional kWh of flexibility provided per day.

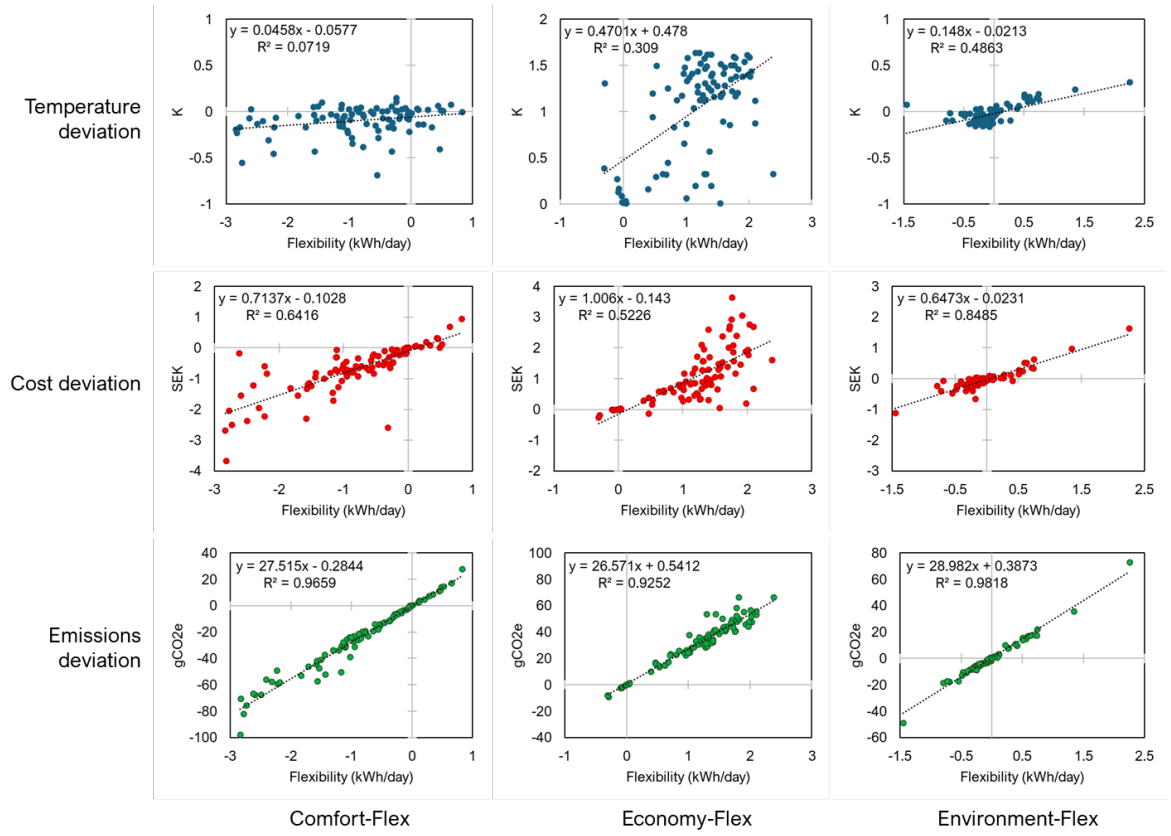

Figure 4. Regression plots of changes in KPIs relative to daily flexibility provision.

The results indicate that the models trained for their respective optimization modes maintain consistent behavior even when flexibility is activated. *Comfort-flex* provides flexibility with the lowest sensitivity in indoor temperature, whereas *economy-flex* and *environment-flex* exhibit approximately 3 and 10 times greater temperature sensitivity per kWh/day, respectively, of activated flexibility. Conversely, the steepest trendline slopes for *economy-flex* and *environment-flex* correspond to their respective optimization KPIs. In particular, the sensitivity of electricity cost to flexibility activation is highest in *economy-flex*, while the sensitivity of CO_2-equivalent emissions is highest in *environment-flex*. This is consistent with the optimization objectives of these modes: activating downward flexibility produces the largest reductions in electricity cost in economy mode and in emissions in environment mode, whereas comfort mode prioritizes maintaining the temperature setpoint.

Finally, the flexibility models generally performed the intended adaptiveness to intraday flexibility requests. A day which had one of the highest flexibility responses is shown in Fig. 5 below. An opportunity for improvement would be to increase the accuracy of the flex responses to fall within the intended bounds and not to under- or over-deliver.

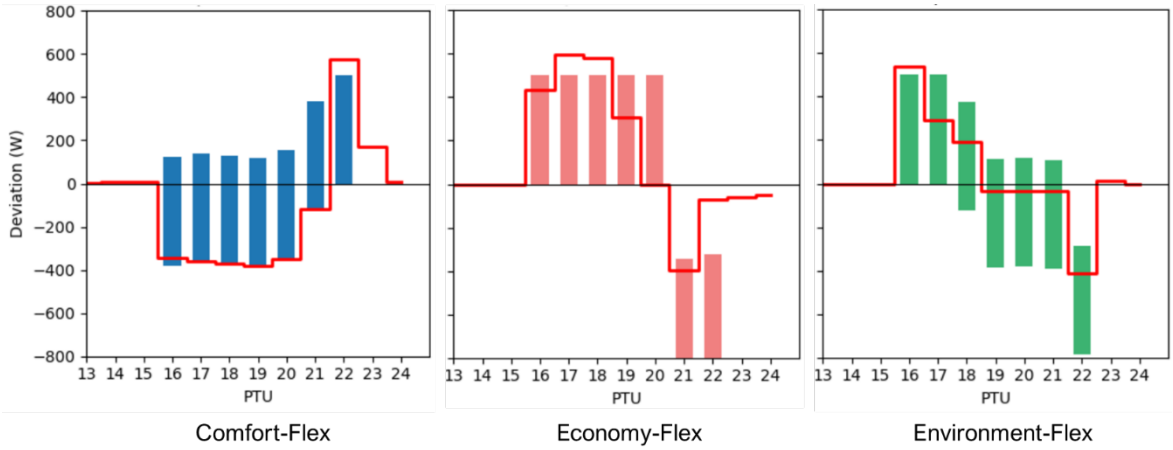

Figure 5. Sample flexibility responses from trained controllers on a selected day (February 24).

4. Conclusions and recommendations

This work demonstrates the significant potential of Deep Reinforcement Learning (DRL) for developing controllers for heat-pump-coupled buildings that enable flexibility while maintaining preferred optimization objectives. Achieving this requires a carefully designed training and evaluation framework, including appropriate selection of state variables, reward design, and training strategy. In this study, curriculum learning was used to train an agent capable of pursuing optimization objectives despite challenges such as thermal inertia and sparse experiences.

More broadly, the proposed approach enables the evaluation and management of trade-offs among user-centric preferences using an RL-based control strategy. The RL-trained models demonstrated their potential to outperform rule-based control (RBC) by learning policies based on a richer representation of system states rather than relying solely on indoor temperature measurements. Furthermore, the models were able to pursue optimization objectives related to cost and environmental impact in addition to comfort, ensuring that financial and environmental objectives do not lead to excessive thermal discomfort. Finally, the results show that activating flexibility introduces trade-offs across multiple KPIs, but the RL-trained controllers were able to keep these deviations within reasonable bounds. If the amount of flexibility that can be activated is lower than requested, the approach for quantifying KPI trade-offs in this work provides a transparent basis for explaining such limitations. In this way, building operators can support decisions on flexibility provision with measurable impacts on comfort, cost, and emissions, enabling more informed dialogue with flexibility requesting parties such as DSOs.

Moving forward, several opportunities for improvement arise. In its current form, the model was trained with only a static temperature setpoint. It would be interesting to train with dynamically changing temperature setpoints, and a more faithful representation of thermal comfort demands. The same goes for other reference state observations like benchmark price, CO2eq emissions, and energy consumption. Moving-average inputs for these may be more suitable. Another limitation that can be addressed later on is the non-inclusion of power tariffs and taxes on the electricity price in both the training and evaluation processes. It would be valuable to see if the same trained models will behave similarly when these are included. Similarly, it would be good to see how forecasts of electricity price, flexibility requests, and weather forecasts will improve model convergence and deployment performance.

To further validate model performance, it would be recommended to evaluate and deploy it with other weather data or expose it to real sensor data and a physical system. Additionally, perhaps the ultimate goal is to have a single model that is adaptive to the modes and optimization objectives instead of several ones that can be chosen based on preferences. Finally, if possible and practical, a parameter sweep for hyperparameters can be attempted to validate or improve the training process.

Acknowledgements

Funded by the European Union's Horizon Europe programme under Grant Agreement no. 101096453 (2023-2025), with continuation of support from Samsung Electronics (2026).

References

[1] USEF Foundation. USEF-The Framework Explained (update 2021); May 2021. Available at: https://www.usef.energy/app/uploads/2021/05/USEF-The-Framework-Explained-update-2021.pdf
[2] Pedersen S.V., Lindahl M., Meesenburg W., Spreitzhofer J., Wernhart M., Rieberer R., Mascherbauer P., Ortmann P., Bakker M. IEA HPT Annex 57: Flexibility by implementation of heat pumps in multi-vector energy systems and thermal networks. Final Report; Report no. HPT-AN57-1; Heat Pump Centre / RISE, Borås (Sweden); August 2023.
[3] Suna D, Totschnig G, Schöniger F, Resch G, Spreitzhofer J, Esterl T. Assessment of flexibility needs and options for a 100% renewable electricity system by 2030 in Austria. Smart Energy 2022;6:1-10.
[4] ENTSO-E & EU DSO Entity. Flexibility Needs Assessment (FNA) Methodology – Type and format of data and methodology for the analysis by TSOs and DSOs. April 2025. Available at: https://consultations.entsoe.eu/system-development/public-consultation-on-flexibility-needs-assessmen/supporting_documents/160425_FNA%20methodology_ENTSOE_DSO%20Entity.pdf
[5] Marijanovic Z, Frank H, Müller-Diemeyer M. Value of short-term heating system flexibility – A case study on the German intraday market. Appl Energy 2022;326:120517.
[6] Aravena C, Riquelme A, Denny E. Money, Comfort or Environment? Priorities and Determinants of Energy Efficiency Investments in Irish Households. J Consum Policy 2016;39:159–186.
[7] De Coninck R, Helsen L. Quantification of flexibility in buildings by cost curves – Methodology and application. Appl Energy 2016;162:653–665.
[8] Nesta Foundation, Centre for Net Zero. HeatFlex: The untapped potential of heat pump flexibility; 18 November 2024. Available at: https://www.nesta.org.uk/documents/3135/HeatFlex_-_the_untapped_potential_of_heat_pump_flexibility.pdf.
[9] Rigaux B, Hamels S, Ovaere M. Heating flexibly with heat pumps: An economic pilot study. *Gentse Economische Inzichten* 2025;18:1-13.
[10] Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge (MA): MIT Press; 2020.
[11] Lissa P, Deane C, Schukat M, Seri F, Keane M, Barrett E. Deep reinforcement learning for home energy management system control. Energy & AI 2021;3:100043.
[12] Langer L, Volling T. A reinforcement learning approach to home energy management for modulating heat pumps and photovoltaic systems. Appl Energy 2022;327:120020.
[13] Bousnina D, Guérassimoff G. Optimal energy management in smart energy systems: A deep reinforcement learning approach and a digital twin case-study. Smart Energy 2024;16:100163.
[14] CATIA-Systems. FMPy: Simulate Functional Mock-up Units (FMUs) in Python. GitHub repository. Available at: https://github.com/CATIA-Systems/FMPy
[15] Raffin A, Hill A, Gleave A, Kanervisto A, Ernestus M, Dormann N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. J Mach Learn Res 2021;22:1-8.
[16] 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;11:3068-82.
[17] Climate.onebuilding.org. WMO Region 6 Europe / SWE – Sweden climate data. Available at: https://climate.onebuilding.org/WMO_Region_6_Europe/SWE_Sweden/index.html
[18] Electricity Maps. Global electricity-grid carbon-intensity data platform. Available at: https://www.electricitymaps.com/
[19] TABULA WebTool. Building-Typology WebTool – http://www.building-typology.eu. Available at: https://webtool.building-typology.eu/?c=se#bm
[20] Simulation Research Group. Modelica Buildings Library – open-source models for building and district energy & control systems. Available at: https://simulationresearch.lbl.gov/modelica/
[21] USEF Foundation. USEF Flexibility Trading Protocol Specifications 1.01; 28 January 2020. Available at: https://www.usef.energy/app/uploads/2020/01/USEF-Flex-Trading-Protocol-Specifications-1.01.pdf

[22] Rolando D, Madani H. Smart Control Strategies for Heat Pump Systems. EffSys Expand Project P18; June 2018. Available at: https://varmtochkallt.se/wp-content/uploads/Projekt/EffsysExpand/P18_Project_Report_final_reviewed.pdf
[23] Jacobs S, Ghane S, Houben PJ, Kabbara Z, Huybrechts T, Hellinckx P, Verhaert I. Improving the learning process of deep reinforcement learning agents operating in collective heating environments. Appl Energy 2025;384:125420.
[24] Laskin M, Srinivasan S, Kushman N. Reinforcement Learning With Augmented Data. In: *Proceedings of the 37th International Conference on Machine Learning* (ICML 2020); 2020.
[25] Payonga LR, Madani H, Stefan M. Curriculum-based Reinforcement Learning for Flexibility Evaluation in Buildings. To appear in: *Proceedings of the IEEE PES International Meeting*; 2026. Currently unpublished.
[26] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347; 2017.