An Hybrid Model-Free Reinforcement Learning Approach for HVAC Control

An Hybrid Model-Free Reinforcement Learning Approach for HVAC Control

Francesco M. Solinas, Andrea Bellagarda, Enrico Macii, Edoardo Patti and Lorenzo Bottaccioli
Politecnico di Torino, Turin, Italy. Email: name.surname@polito.it

***Abstract*—Traditional Heating Ventilation and Air Conditioning (HVAC) systems are extremely energy draining appliances, and their use is ever increasing with urbanisation. For this reason, strong research effort has been put in the development of novel control strategies for the optimal management of HVAC systems, aiming at reducing energy consumption without affecting thermal comfort. In this paper, we propose an hybrid model-free Reinforcement Learning approach for HVAC control able to optimise both energy consumption or users comfort. Our methodology is compared with two baseline solutions in literature based on an EnergyPlus controller and a Model Predictive Control. Results show that our methodology can outperform both baselines in terms of energy consumption reduction or thermal comfort optimisation, given that either of the two objectives is appropriately chosen during the training and the hyperparameters selection phase.**

***Index Terms*—Artificial Intelligence, HVAC optimisation, Reinforcement Learning, Smart Buildings**

I. Introduction

Traditional Heating Ventilation and Air Conditioning (HVAC) systems are one of the most energy demanding appliances in our buildings [1]. Such systems, are traditionally managed thanks to a rule-based approach that is based on two steps: *1)* the definition of a setpoint and *2)* the usage of Proportional Integrated Derivative (PID) control to track the setpoint temperature [2]. In the last years a strong research effort has been put in the application of novel control strategies [3] able to reduce energy consumption without decreasing thermal comfort and effectively replacing classic PID controller methods. Among these, some of the most effective technologies are based on Model Predictive Control (MPC) [4] and Reinforcement Learning (RL) [5].

An MPC implementation can be summarised with three main phases: i) a "Modeling" phase, which consists in the development and identification of models capable of fully characterising the thermal and energy dynamics of buildings and systems; ii) a "Prediction" phase, in which a trajectory of future states of the system is built thanks to the resort to the previously constructed model; iii) a "Control" phase, in which the optimisation problem is solved. The major challenge of MPC is that it is labor-intensive and requires expertise to use [5]. Serale et al. [6] illustrates the potential benefits of MPC applications to building temperature control and specifically HVAC energy management. In [7] GNU-RL is presented, a hybrid approach in which a differentiable MPC [8] is coupled with a Proximal Policy Algorithm (PPO) [9] for HVAC control.

Among RL techniques, it is possible to identify two major categories: model-based RL and model-free RL. In model-based RL the characteristics of the environment are learned to find the optimal policy. The model could be: i) a *white-box model*, in the case of building temperature control, this would be normally simulated in the EnergyPlus (E+) software; ii) a complete *black-box model*, such as a neural network, which learn the system dynamics solely from data; iii) or a *grey-box model*, such as those characterised by exploiting Kalman filters [10]. Model-based RL is somewhat similar to the MPC technique, as it requires a model, a representation of the environment in order to find the optimal control policy. Zhang et al. [11] present an application of Monte-Carlo-Tree-Search (MCTS), a model-based RL technique, to HVAC optimisation and a neural network representation of the system model. Differently than MPC and model-based RL, model-free RL avoids the time-consuming process to represent a model of the system under analysis. Indeed, it learns the optimal control policy by direct interaction with the environment.

In its simpler form, a pure model-free RL controller should be trained through trial-and-error interaction directly with the building. Which means, that the controller has to apply the policy in the environment to evaluate the effects of the chosen strategy. Clearly, this approach is not viable as it would lead to sub-optimal scheduling of HVAC system during the training phase of the agent, thus generating discomfort in the occupants until a good policy is learned. In a real-world scenario, it is unrealistic that a building operator would allow an RL controller to learn by trial-and-error a policy on the real building HVAC system. For this reason, model-free RL approaches to HVAC control, and building temperature control in general, rely on white-box simulation as the test-bed for learning a policy in a virtual environment, as representative as possible of the real-world building and its dynamics. However, white-box simulations have to be designed from scratch for each new building where the new optimisation strategy needs to be applied to. They are long to design and computationally expensive to perform. Model-free RL approaches thus suffers from the resort to such simulations [5].

This notwithstanding, many effective approach of tackling building temperature control has been performed using RL and systems simulations. In [12], a Gradient Bandit algorithm is presented for Peak-Shaving in district heating networks, adopting a simulation for the thermodynamics of the district heating network and for modeling the individual thermal response of the individual buildings. In [13], the heat water

supply of a building is optimised through a Deep Q Network (DQN) [14], and E+ is used as the training environment for the RL agent. In [15], a policy gradient algorithm is adopted to optimise over the energy efficiency and thermal comfort of a building. HVAC systems optimisation through RL has been also presented in [16], where RL agents showed better performance over a programmable controller, and [17], where Deep RL and the simulation software E+ are adopted to optimise over energy consumption while respecting a thermal comfort threshold.

This paper aims at overcoming the described shortcomings of model-free RL application to building temperature control by introducing a novel approach to energy optimisation in HVAC systems. Our solution is based on a hybrid-approach, where a model-free RL algorithm is paired with a black-box system identification model, which substitutes the white-box simulation of the building dynamics. In this fashion, our RL agent is free to interact with the environment, performing repeated actions as if it was acting on a real building HVAC system. Our proposed methodology is therefore more flexible than the traditional approach, as it is able to tackle HVAC optimisation only from historical building data, needed for constructing a reliable system identification, and can do without costly and time-consuming white-box simulations. The adoption of a model-free Reinforcement Learning algorithm, and black-box model for the system identification, makes our approach hybrid and able to be easily replicable on any building, given the simple availability of some historical data, and without the need of modelling the entire building dynamics in a simulator such as E+. The authors also contributed to the extension of an OpenAI Gym environment E+ implementation [18], presented in [19], used for real-time testing of the developed methodology.

The rest of the paper is structured as follows: Section II describes the proposed methodology, detailing the features of the system identification phase and the adopted reinforcement learning algorithm. Section III discusses our experimental results providing a comparison with two baseline solutions in literature based on an E+ controller and a Model Predictive Control [7]. Lastly, Section IV reports our concluding remarks.

II. Methodology

Our solution for the energy optimisation in HVAC systems is based on a two-fold approach: at first, a system identification model is developed through Supervised Learning on historical data about a building energy supply and contextual weather conditions; then, a model-free Reinforcement Learning algorithm is employed to optimise over the energy distribution in said building.

The use of a Machine Learning based system as the identification model for the underlying building dynamics helps mitigating the shortcomings of a pure model-free approach. Instead of performing actions in a real-world scenario, represented by the building itself, or in a computationally costly white-box simulation, our RL agent is able to freely interact with this black-box environment, and is therefore able to train appropriately as if it was interacting with the real-world environment, saving computational and modeling costs. As

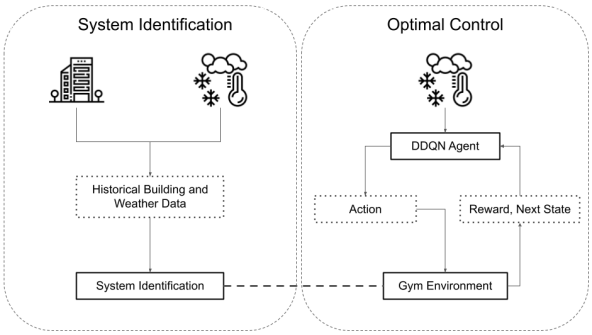

Fig. 1: The proposed workflow

shown in Fig. 1, at first a System Identification training phase will be performed on historical data. Then, the resulting trained network will be used for making effective inferences about the building thermal dynamics, estimating the variations of temperature in the building given outdoor and indoor conditions and the action taken by the RL agent. When the training phase is completed, the System Identification model will be used by the Reinforcement Learning algorithm as a real-time environment, modeled in a similar fashion to an OpenAI Gym environment [20], as a replacement for the real-world building. Thanks to this, the agent is able to receive real-time feedback for the actions taken, calculating the corresponding reward and learning the optimal control policy.

A. System Identification

The first phase of our proposed solution is based on a System Identification model, similar to what is presented in [7]. A black-box approach is adopted, in which a Supervised Learning regression algorithm is trained on buildings and weather data, in order to properly estimate the building thermal response to different energy provisions and various weather conditions.

The input for the training phase of the model is represented by the building historical data gathered on a normal heating cycle. More specifically, the Supervised Learning algorithm takes as input the initial indoor air temperature of the building, the action performed by the HVAC controller and a set of weather conditions that have influence on the heating dynamics, and maps those input variables into the resulting indoor air temperature. This prediction is then compared to the actual resulting temperature, a mean-squared-error loss Eq. 1 is calculate and a gradient descent step is performed accordingly.

$$\mathcal{L}_\theta = \sum_t (x_t - \hat{x}_t)^2 \tag{1}$$

Instead of using a Neural Network as a non-linear approximator for the underlying thermal dynamics of the process, the proposed System Identification is based on a simple linear

function f_θ and a set of parameters $\theta = [A, B, D]$ as showed in Eq. 2:

$$f_\theta(x_t, u_t, d_t) = \hat{x}_{t+1} = Ax_t + Bu_t + Dd_t \quad (2)$$

The input variable for the system identification model, namely the set $[x, u, d]$, includes the indoor temperature x ($^\circ C$), the control action u ($^\circ C$) and the set of disturbances d. The latter consists of: the outdoor temperature $x_{outdoor}$ ($^\circ C$), the outdoor relative humidity $RH_{outdoor}$ ($\%$), the wind speed $Wind_{speed}$ (m/s), the wind direction $Wind_{dir}$ ($degrees$), the diffracted solar radiation Rad_{diff} (W/m^2), the direct solar radiation Rad_{direct} (W/m^2) and the current occupancy flag Occ_{flag} (Boolean).

Alg. 1 describes the algorithm adopted for the System Identification. The algorithm is a simple instance of a Supervised Learning algorithm, where the prediction of the function f_θ is compared with the actual next temperature x_{t+1} as observed by the building historical data. Then a mean-square-error is calculated and the loss is back-propagated through gradient descent in order to train the set of parameters $\theta = [A, B, D]$.

Algorithm 1 System Identification

1: inputs: learning rate α, maximum epochs $Epochs_{max}$ and steps $Steps_{max}$
2: Initialise randomly the parameters $\theta = [A, B, D]$
3: **while** $Epoch < Epoch_{max}$ **do**
4: **while** $Steps < Steps_{max}$ **do**
5: $\hat{x}_{t+1} = f_\theta(x_t, u_t, d_t)$
6: **end while**
7: $\theta \leftarrow \theta - \alpha \nabla \mathcal{L}_\theta$
8: **end while**

B. Optimal Control

The second stage of our proposed methodology consists in the deployment of a Reinforcement Learning (RL) agent, responsible for the actual energy optimisation of the HVAC system. RL problems are represented as Markov Decision Problems (MDP), which are constituted by the tuple (s, u, r, s'). An agent takes a control action u in an initial state s, applying it to the environment transition function f_θ and receiving a reward r and a new state s'. The RL agents has to learn the best policy that enables it to take the optimal control actions u in every state s of the environment. In what follows, the elements of the HVAC optimisation problem represented as a RL problem above is described.

The objective of the RL agent is that of maximising the expected reward as provided by a reward function $R(s, u) \rightarrow r$ which takes as input a state s and a control action u and provides the reward r. The objective of our optimisation task consists on reducing as much as possible the energy consumption in HVAC systems, while maintaining the indoor air temperature to a comfort threshold, i.e. as close as possible to the given temperature setpoint. In order to accomplish this, the reward r has been chosen to be equal to the cost of: the control action u_t plus the squared distance of the resulting temperature x_t from the given setpoint $x_{setpoint}$. The weights β and ρ determine how important is one side of the equation relatively to the other. More specifically, if β is higher than ρ, the agent will be more inclined to keep the indoor temperature closer x_t to the given setpoint $x_{setpoint}$. On the other hand, higher values of ρ, will push the agent to reduce more significantly the energy consumption. Note that the setpoint $x_{setpoint}$ assume different values during occupied or vacancy periods; accordingly, two different values for β can be provided.

$$R(s, u) = r = -(\beta \cdot (x_t - x_{setpoint})^2 + \rho \cdot u_t) \quad (3)$$

The state s, the input to the RL agent, is represented by all the relevant pieces of information about the building internal and outdoor conditions. The input state has to be as informative as possible about the underlying system dynamics and about the conditions that influences the reward function, which has to be maximised.

More specifically, state s includes: $x_{setpoint,t} - x_t$ ($^\circ C$), which is the difference between the setpoint and the internal temperature, taken with a 4 periods lag - so four times from $t-4$ to t; the difference between the setpoint and the outdoor air temperature $x_{setpoint,t} - x_{outdoor,t}$ ($^\circ C$); the outdoor relative humidity $RH_{outdoor}$ ($\%$); the wind speed $Wind_{speed}$ (m/s); the wind direction $Wind_{dir}$ ($degrees$); the diffracted solar radiation Rad_{diff} (W/m^2);the direct solar radiation Rad_{direct} (W/m^2); the number of hours before the start or the end of the next or the ongoing occupancy period, Occ_{start} and Occ_{end} (h) respectively. All inputs are pre-processed according to the $MinMax$ normalisation rule.

The state transition function f_θ has already been presented in Eq. 2, as it is the function resulting from the System Identification and the Supervised Learning training phase on the building historical data. f_θ correlates the initial building temperature x_t, and the outdoor weather conditions d_t, with the control action u_t, giving an effective estimate over the resulting internal temperature x_{t+1} and thus the next state s'.

The action-space, namely the set of possible actions that the agent can apply to the environment, is discrete and consists in the four possible different temperature degrees [0,2,4,5.5] at which the air is heated before being supplied to the room. A limited action-space has been chosen, since allowing the agent to perform a more fine-grained range of actions, or even a continuous one, would not increase the agent's effectiveness, but would drastically increase the problem complexity [21].

The proposed algorithm is a Double Deep Q Network (DDQN) [22]. This algorithm, fully described in Alg. 2, presents better converging properties than the standard Deep Q Network (DQN), without significantly increasing the complexity of the model.

As the standard DQN [14], the algorithm is based on a Neural Network, Q_ω, that acts as a function approximator for the Q values, which are a metrics of estimate for the quality of a certain action in a certain state. The more expected reward r an action u is estimated to yield in a certain state s at time t, the higher its associated Q value, $Q(s_t, u_t; \omega)$, will be. A

Algorithm 2 DDQN

Random initialise parameters ω of network Q_ω, and parameters $\omega' \leftarrow \omega$ of target network $Q_{\omega'}$
Initialise replay memory D, learning rate α and target network update parameter τ
while $Episode < EP_{max}$ **do**
  while $Steps < Steps_{max}$ **do**
    With probability ϵ perform random action u_t, otherwise observe state s_t and perform action $u_{t+1} = max\ Q(s_t, u_t; \omega)$
    Perform action u_{t+1} in the environment, get the reward r_{t+1}, calculate the new temperature $x_{t+1} = f_\theta(s_t, u_t)$ and observe next state $s' = [x_{t+1}, d_{t+1}]$
    Store (s, u, r, s') in the replay memory D
    Update $\epsilon \leftarrow (\epsilon_{init} - \epsilon_{fin})/Exploration_{Steps}$
  end while
  while $Update\_Steps < Update\_Steps_{max}$ **do**
    Sample a batch (s_t, u_t, r_t, s'_t) of size Bs from memory D
    Compute target Q value: $Q^*(s_t, u_t) = r_t + \gamma \cdot Q_\omega(s_{t+1}, argmax_u Q_{\omega'}(s_{t+1}, u_t))$
    Perform gradient descent step on $(Q^*(s_t, u_t) - Q_\omega(s_t, u_t))^2$
    Update $Q_{\omega'}$ parameters: $\omega' \leftarrow \tau \cdot \omega + (1 - \tau) * \omega'$
  end while
end while

common issue with the standard DQN is represented by biased estimates, as the neural network is updated based on its very same estimates of the quality of a certain action. In order to disentangle the network from these biased estimates, the Double DQN introduces a second target neural network $Q_{\omega'}$. The primary network Q_ω updates the other every a certain number of steps (or continuously at a smaller rate). The target network is used for action selection while the primary network for action evaluation, in the computation for the target values Q^* as shown in Eq. 4. $Q^*(s_t, u_t)$ is the reference term used for the Q values update, and in the DQN methodology, it represents the $true$ Q values estimation, namely the true value of taking action u_t in state s_t.

$$Q^*(s_t, u_t) = r_t + \gamma \cdot Q_\omega(s_{t+1}, argmax_u Q_{\omega'}(s_{t+1}, u_t)) \quad (4)$$

Factor γ plays a fundamental role in this optimisation process. γ determines the horizon estimation of the DQN agent. For lower values of γ, the agent will favour the exploitation of short-term reward, as the relative weight of the immediate reward r_t in Eq. 4 will be higher. For the same reason, higher values of γ will push the agent to rather seek long-term reward, represented by the $Q_\omega(s_{t+1}, argmax_u Q_{\omega'}(s_{t+1}, u_t))$ estimate in the target $Q*$ computation.

III. Results

In this section, results for the system identification and for the HVAC optimal control algorithm are presented. The latter is compared against an E+ standard controller and a literature based solution, namely the GNU-RL algorithm [7]. For this reason, the same E+ controller and 5-zone building, retrievable here [23], have been used as a case study for a fair comparison. Despite being a 5-zone building, in all experiments the five indoor air temperatures are averaged and the building is treated as if only one indoor average temperature is relevant. Three months building and weather historical data used for the System Identification phase and the training of the DDQN agent are generated in E+ based on the $pittsburgh\_TMY2.epw$ weather file. The testing of the DDQN agent and the two baselines, is performed on the same building, on the $pittsburgh\_TMY3.epw$ weather file. In order to test the three methods, a real-time implementation of E+ as a OpenAI Gym environment was required. The adopted implementation was developed in [19] and retrievable at [18]. This Gym-Eplus implementation has been extended by the authors of this paper, in order to upgrade the E+ versions compatibility.

A. System Identification

In this section, results for the System Identification are presented. As mentioned above, the training dataset is constituted by 3 months of data gathered from the E+ implementation of a 5-Zone building HVAC system and the $pittsburgh\_TMY2.epw$ weather file.

Fig. 2 shows the results in terms of mean-square-error in the first 100 epochs of training. It can be observed that the proposed system identification is able to predict the next indoor air temperature of the target building with a deviation as low as $0.06\ ^\circ C$ on average.

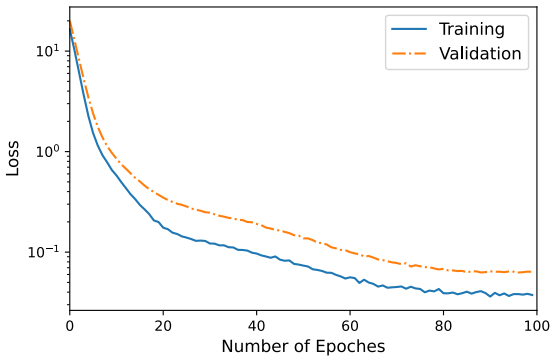

Fig. 2: System Identification training results

B. Optimal Control

In this section, the proposed methodology is experimentally compared with an E+ baseline controller and with the literature solution in [7]. The target environment is represented by a simple building with a HVAC system, modeled in E+. The historical data are gathered from a 3-months long simulation in E+. The building is treated as having only one thermal zone for the sake of simplicity.

As discussed in Section II, in our proposed solution there is no need to perform a simulation step in E+ every time the agent takes a new action. Instead, our solution relies on the trained system identification model, which enables a reliable estimate of the building thermal dynamics without recurring to costly and time-consuming white-box simulations.

In order to show the higher flexibility and large range of accomplished results, multiple simulations for various value of γ are presented for our solution, namely 0.8, 0.9 and 0.99 are the tested values. The hyperparameter γ is fundamental in mediating between the maximisation of the immediate reward and the long-term expected reward. Higher values of

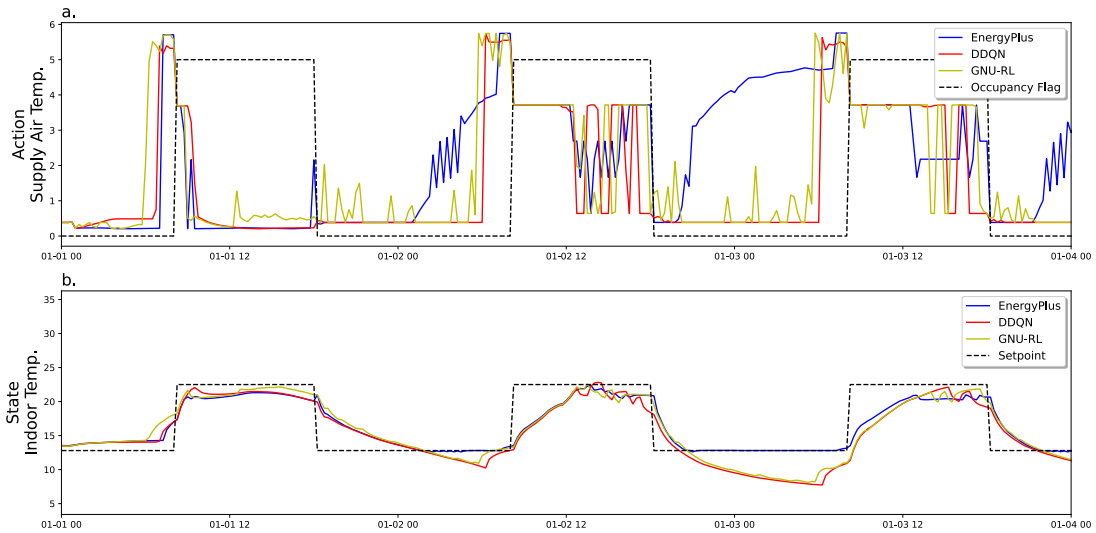

Fig. 3: Actions comparison for the three approaches and the resulting indoor temperature

γ push the agent to seek a long-term reward, while lower values compel the agent to look for more immediate compensation for its actions. In our HVAC system optimisation scenario, γ is responsible for the trade-off between a short-term reward, namely a lower energy consumption, and a long-term compensation translated into a smaller distance to the temperature setpoint and a higher thermal comfort for the building occupants. The indoor setpoint $x_{setpoint}$ is set for all experiments to be 12.8 $°C$ when the building is vacant and 22.5 $°C$ for the building occupancy periods. β is set to 0.05 for vacancy and 2 for occupancy periods, while ρ is kept constant at 1. The DDQN algorithm hyperparameters are set as follows: the maximum number of steps for each episode $Steps_{max}$ is 8732; the number of steps $Exploration_{Steps}$ in which the agent explores the action space is $3 \cdot 10^5$; in this period, the probability ϵ of taking a random action starts at 1 and ends at 0.1; the network update parameter τ is 0.01.

The adopted neural network, implemented in PyTorch, that acts as a function approximator for the Q values in the proposed DDQN algorithm is a fully connected network made of 3 layers with 64 neurons each. The learning rate of the neural network is set to 0.001 and the batch size for the training update is 32.

As mentioned before, our proposed solution is compared with two baseline solutions in literature based on an E+ controller and a Model Predictive Control [7]. GNU-RL solution is based on a Model Predictive Control (MPC) algorithm, which is responsible for the optimisation of the same cost function that has been presented in Eq. 3. The three approaches are compared with each other in a simulation performed on E+ on the $pittsburgh\_TMY3.epw$ weather file, for a period of 3 months, on the basis of the overall energy expenditure and the average Predicted Percentage of Dissatisfaction (PPD). PPD is a widely used metrics for measuring the thermal comfort of building occupants. It ranges from an optimal value of 5% up to the worst case scenario, in which the whole population is dissatisfied by the thermal conditions. The PPD is deemed acceptable as long as it stays under 20%.

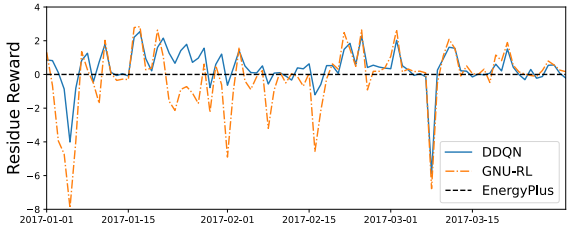

Fig. 4: Overall comparison in terms of residue reward

A summary comparison of the three approaches can be observed in Tab. I, where the E+ controller, the GNU-RL algorithm and different values of γ for our proposed solution are compared according to the above-mentioned metrics.

It can be noted that the proposed solution outperforms the two baselines in both the average PPD and the overall energy consumption. In order to do so, however, the DDQN agent needs to be trained with a different value for γ so that it can privilege one aspect over the other. More specifically, for lower values of γ, the agent learns to favour the immediate reward, greatly reducing the energy consumption, while slightly increasing the PPD. Higher values of γ, instead, tends to favour the long-term expected reward, which is represented by the temperature distance to the indoor setpoint, at a higher energy consumption cost. This happens because the penalty for energy consumption of Eq. 3 applies during both the occupancy and vacancy period of the building, representing a constant cost that the agent has to face. On the other hand, the term related to the thermal comfort only determines a significant influence on the reward during occupancy period, so for only 8 hours a day. In order to guarantee an appropriate thermal comfort, however,

TABLE I: Results Comparison

	E+	*GNU-RL*	$DDQN_{\gamma=0.8}$	$DDQN_{\gamma=0.9}$	$DDQN_{\gamma=0.99}$
PPD	17.75%	16.46%	16.61%	17.31%	**15.45%**
HVAC Power	4413kWh	4215kWh	4097kWh	**4093kWh**	5220kWh
Coil Power	7482kWh	7421kWh	7248kWh	**7228kWh**	8381kWh

the agent has to start the heating of the building long before the occupancy period starts, as it can be observed in Fig. 3b. All things considered, this causes a greedy agent, at lower values of γ, to care more about the immediate reward, and thus about the energy consumption, while it makes a more cautious agent, at higher values of γ, look for greater future reward, and thus for the thermal comfort of the building occupants.

Fig. 3a shows a comparison of the control actions taken by our proposed solution ($\gamma = 0.8$) and the two baselines on three typical winter days. Consequent effects on the indoor temperature variations, and the setpoint, are shown in Fig. 3b.

Fig. 4 shows the overall performance of the proposed solution ($\gamma = 0.9$) and that of GNU-RL compared with the E+ baseline in terms of residue reward, which is the difference between the reward, as defined in Eq. 3, obtained by the E+ controller and the reward obtained by the two compared methods, GNU-RL and our solution. It can be noted how our solution generally outperforms both baselines in the entire investigation period of 3 months.

IV. Conclusion

This paper presented a novel approach to HVAC system optimal control, implementing a System Identification phase and a model-free RL algorithm. The system identification phase, based on a supervised learning algorithm, is able to precisely estimate the building thermal dynamics, allowing the RL agent to freely interact with such environment without the need of direct, real-world interaction with a real building or with costly and time-consuming white-box simulations. Results have shown that the proposed methodology, based on a Double Deep Q Network, is able to outperform the two discussed baseline, an E+ controller and the Gnu-RL algorithm [7]. By simply varying the hyperparameter γ during the training phase of the RL agent, it is possible to optimise over the overall energy consumption or the internal comfort according to the PPD metrics. Future research can expand the presented work in two directions: on the one hand, a more complex problem can be tackled, in which the learning agent has to take multiple simultaneous actions to optimize over a series of different thermal zones at once; on the other hand, given the accurate model of the environment presented here, a fully RL model-based approach could be investigated and compared with the proposed hybrid methodology.

References

[1] M. Kharseh, L. Altorkmany, M. Al-Khawaj, and F. Hassani, "Warming impact on energy use of hvac system in buildings of different thermal qualities and in different climates," *Energy Conversion and Management*, vol. 81, pp. 106–111, 2014.

[2] K. Mařík, J. Rojíček, P. Stluka, and J. Vass, "Advanced hvac control: Theory vs. reality," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 3108–3113, 2011.

[3] F. Belic, Z. Hocenski, and D. Sliskovic, "Hvac control methods-a review," in *In Proc. of ICSTCC 2015*. IEEE, 2015, pp. 679–686.

[4] J. Drgoňa, J. Arroyo, I. C. Figueroa, D. Blum, K. Arendt, D. Kim, E. P. Ollé, J. Oravec, M. Wetter, D. L. Vrabie *et al.*, "All you need to know about model predictive control for buildings," *Annual Reviews in Control*, 2020.

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

[6] G. Serale, M. Fiorentini, A. Capozzoli, D. Bernardini, and A. Bemporad, "Model predictive control (mpc) for enhancing building and hvac system energy efficiency: Problem formulation, applications and opportunities," *Energies*, vol. 11, no. 3, p. 631, 2018.

[7] B. Chen, Z. Cai, and M. Bergés, "Gnu-rl: A precocial reinforcement learning solution for building hvac control using a differentiable mpc policy," in *In Proc. of BuildSys '19*, 2019, pp. 316–325.

[8] B. Amos, I. D. J. Rodriguez, J. Sacks, B. Boots, and J. Z. Kolter, "Differentiable mpc for end-to-end planning and control," *arXiv preprint arXiv:1810.13400*, 2018.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[10] M. Massano, E. Patti, E. Macii, A. Acquaviva, and L. Bottaccioli, "An online grey-box model based on unscented kalman filter to predict temperature profiles in smart buildings," *Energies*, vol. 13, no. 8, p. 2097, 2020.

[11] C. Zhang, S. R. Kuppannagari, R. Kannan, and V. K. Prasanna, "Building hvac scheduling using reinforcement learning via neural network based model approximation," in *In Proc. of BuildSys '19*, ser. BuildSys '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 287–296. [Online]. Available: https://doi.org/10.1145/3360322.3360861

[12] F. M. Solinas, L. Bottaccioli, E. Guelpa, V. Verda, and E. Patti, "Peak shaving in district heating exploiting reinforcement learning and agent-based modelling," *Engineering Applications of Artificial Intelligence*, vol. 102, p. 104235, 2021.

[13] S. Brandi, M. S. Piscitelli, M. Martellacci, and A. Capozzoli, "Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings," *Energy and Buildings*, vol. 224, p. 110225, 2020.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[15] R. Jia, M. Jin, K. Sun, T. Hong, and C. Spanos, "Advanced building control via deep reinforcement learning," *Energy Procedia*, vol. 158, pp. 6158–6163, 2019.

[16] E. Barrett and S. Linder, "Autonomous hvac control, a reinforcement learning approach," in *In Proc. of ECML PKDD 2015*. Springer, 2015, pp. 3–19.

[17] T. Wei, Y. Wang, and Q. Zhu, "Deep reinforcement learning for building hvac control," in *In Proc. of DAC 2017*, 2017, pp. 1–6.

[18] Z. Zhang and K. P. Lam, "Gym-eplus," https://github.com/zhangzhizza/Gym-Eplus, 2019.

[19] Z. Zhang and K. P. Lam, "Practical implementation and evaluation of deep reinforcement learning control for a radiant heating system," in *In Proc. of BuildSys '18*, 2018, pp. 148–157.

[20] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[21] G. Dulac-Arnold, R. Evans, P. Sunehag, and B. Coppin, "Reinforcement learning in large discrete action spaces," *CoRR*, vol. abs/1512.07679, 2015. [Online]. Available: http://arxiv.org/abs/1512.07679

[22] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *In Proc. of AAAI 2016*, vol. 30, no. 1, 2016.

[23] B. Chen, Z. Cai, and M. Bergés, "Gnu-rl," https://github.com/INFERLab/Gnu-RL, 2019.