Domain Randomization for Demand Response of an Electric Water Heater

Thijs Peirelinck, *Graduate Student Member, IEEE*, Chris Hermans, Fred Spiessens, and Geert Deconinck, *Senior Member, IEEE*

***Abstract*—Thermostatically Controlled Loads (TCLs) provide a source of demand flexibility, and are often considered a good source for Demand Response (DR) applications. Due to their heterogeneity, and as such a lack of dynamics models, Reinforcement Learning (RL) is often used to exploit this flexibility. Unfortunately, RL requires exploratory interaction with the TCL, resulting in a period of potential discomfort for the users. We present an approach to reduce this exploratory time by pre-training the RL-agent. Domain randomization is used to facilitate knowledge transfer. We evaluate the pre-training potential in a DR energy arbitrage scenario with an Electric Water Heater (EWH). Our experiments show that a priori knowledge about EWH dynamics can be used to initialize and improve the control policy. In our experiments, pre-training attributes to 8.8 % additional cost savings, compared to starting from scratch.**

***Index Terms*—Demand response, electric water heater, reinforcement learning, domain randomization.**

I. Introduction

DEMAND response programs aim to exploit consumption flexibility. A common challenge in DR is developing scalable control algorithms. In the past, residential TCLs were identified as promising appliances for DR, mainly due to their inherent flexibility emerging from their decoupled electricity and heat demand [1]. But, a wide variety of TCLs exist, thus amplifying the need for scalable control [2].

RL has shown promising results in several different DR applications. Four examples are considered here. In a first application, Ruelens *et al.* [1] illustrate that a RL method called Fitted Q-Iteration (FQI) reduces the electricity consumption cost of an EWH by 24 %, when charged with Belgian day-ahead electricity prices. Similar results have been obtained for space-heating [3], [4]. In a second application, De Somer *et al.* [5] use EWH storage and FQI for local photovoltaic (PV) self-consumption. During their four month real-life demonstration with six residential buildings they observed an increase of 20% in total PV self-consumption. In a third application, Liu *et al.* [2] use an aggregation of EWHs to balance wind power generation. Our fourth and final example of DR applications consists of the work of Kazmi *et al.* [6], in which the authors employ RL to increase energy efficiency of EWHs. While maintaining occupant comfort their controller reduced energy consumption for hot water production by almost 20%.

While all the results mentioned above are promising, they share the same drawbacks. RL has an unavoidable initialization/learning phase, where the controller inevitably takes random actions. Ruelens *et al.* [7] acknowledge this drawback and, building upon their earlier work [1], they propose a policy adjustment method based on expert domain knowledge. Contrary to the approach of using domain knowledge, Lampe and Riedmiller [8] introduced Model-Assisted Fitted Q-Iteration (MAFQI). In this method, a model is trained on observed data and virtual trajectories are added to the training set, reducing interaction time needed to reach reasonable policies. The experimental analysis of Costanzo *et al.* [9] showed MAFQI can obtain sensible DR policies after 10 days and, these policies reach a nearly optimal performance after 20 days.

Recent breakthroughs in RL indicate we can avoid learning a model from observations in case we have prior knowledge about the system's dynamics. Tobin *et al.* [10] showed it is possible to transfer Neural Networks (NNs) from simulation (source domain) to practice (target domain). They proposed domain randomization, i.e., randomize the source domain in certain parameters. Further research [11] has shown domain randomization greatly reduces target domain training time and policies are capable of adapting to unfamiliar target-system dynamics. It is unclear if these results are applicable to the DR setting.

A wide variety of EWH buffer models has recently been presented [12]–[14]. We aim to exploit both the modelling and RL research. Our objective is two-fold: apply RL for EWH control *and* minimize initialization time. Our main contribution consists of combining the usage of these buffer models with domain randomization, in order to reduce initialization time when applying RL in a DR setting. The aim of this work is thus to verify that the combination of buffer models, domain randomization and RL is fruitful for DR. As a first step, both our source and target domain are simulated. However, as the lab-validated target domain model is more

Manuscript received March 12, 2020; revised June 3, 2020 and August 7, 2020; accepted August 27, 2020. Date of publication September 21, 2020; date of current version February 26, 2021. This work was supported by the Flemish Institute for Technological Research (VITO). Paper no. TSG-00362-2020. *(Corresponding author: Thijs Peirelinck.)*

Thijs Peirelinck and Geert Deconinck are with ESAT/Electa, KU Leuven, 3001 Leuven, Belgium, and also with AMO, EnergyVille, 3600 Genk, Belgium (e-mail: thijs.peirelinck@kuleuven.be).

Chris Hermans and Fred Spiessens are with AMO, EnergyVille, 3600 Genk, Belgium, and also with AMO, Flemish Institute for Technological Research (VITO), 2400 Mol, Belgium.

Digital Object Identifier 10.1109/TSG.2020.3024656

TABLE I
BUFFER MODEL PARAMETERS

Name	Explanation	Value	Unit
C_{boiler}	Buffer capacitance	798142	J/K
U	Buffer thermal transmittance	0.75	W/(m^{2}K)
C_{pw}	Water specific heat capacitance	4182	J/(kgK)
h	Buffer height	1.2	m
d	Buffer diameter	0.45	m
A_s	Buffer side surface	1.7	m^2
A_t	Buffer top/bottom surface	0.16	m^2
V	Buffer volume	190.85	kg
P_r	Rated power	2400	W
T_{amb}	Ambient temperature	20	°C
T_{iw}	Inlet water temperature	17	°C
T_{min}	Minimal buffer temperature	55	°C
T_{max}	Maximum water temperature	70	°C
T_{asked}	Asked output water temperature	45	°C

TABLE II
BUFFER MODEL VARIABLES

Name	Explanation	Unit
$\dot{Q}_{\text{heat}}$	Heat flow rate heating element	J/s
$\dot{m}_w$	Hot water tap demand	kg/s
T_i	Water temperature (of layer i)	°C
SoC	State of Charge	[-]

elaborate than the source domain model(s), this results in a clear modelling difference.

In Section II we present the different buffer models used throughout this article. Section III begins by laying out the problem formulation and, thereafter, looks at the RL algorithms. Section IV is concerned with the methodology. Consequently, in Section V, we present the experiments and their findings. In Section VI, we conclude this work and indicate future work directions.

II. ELECTRIC WATER HEATER MODELS

Throughout this work we use three buffer models. We will start by introducing them and end with using them for calculating water temperature, which clearly shows where each model resembles or differs from the others.

All three models rely on the heat balance equation. As a result, they share mostly the same parameters and variables, summarized in Tables I and II. The buffer has rated power P_r and tank volume V. While hot water is tapped at the top, it is replaced by cold water at the bottom, where the heating element is located. Total boiler capacitance C_{boiler} equals $V \cdot C_{pw}$, with C_{pw} the specific heat capacitance of water.

Tap demand $\dot{m}_w$ has been obtained from hot water tap profiles at a 5 minute resolution [15]. The profile's daily average Domestic Hot Water (DHW) consumption equals 189 l/day. Edwards *et al.* [15] state it corresponds to an average consumption level of a household mainly demanding hot water in the evening.

All buffers include a hysteresis controller, which preserves user comfort and turns the heating element on when water temperature drops below T_{min} and off when it raises above T_{max}. We assume this controller cannot be by-passed. External control is only possible when the water is in between these temperature limits. Requested T_{asked}, inlet T_{iw} water and ambient T_{amb} temperature have been assumed constant.

We assume the buffer's internal State of Charge (*SoC*) [14] can be calculated from the available measurements. The *SoC* expresses a measure of the buffer's energy content, relative to its minimal and maximal allowed energy stored. Hence, $SoC = 0$ occurs when the whole water content is at temperature T_{min} and $SoC = 1$ when it is at T_{max}. Water colder than T_{min} is not considered.

A. Uniform Buffer Model

We use the buffer model as presented by Farooq *et al.* [12]. This model describes the heat balance of a mass of water, with temperature T_L. It models three processes: heat supplied by the heating element, transfer of heat by the inlet water and losses to the environment. The model is given by equation (1).

$$\frac{dT_L}{dt} = \frac{1}{C_{\text{boiler}}}\big(\dot{Q}_{\text{heat}} + \dot{m}_w C_{pw}(T_{iw} - T_L) + U(A_s + 2A_t)(T_{\text{amb}} - T_L)\big) \quad (1)$$

All symbols are defined in Tables I and II. Since the water content of the buffer has a uniform temperature *SoC* is defined as in (2).

$$SoC = \frac{T_L - T_{\text{min}}}{T_{\text{max}} - T_{\text{min}}}. \quad (2)$$

B. Two Mass Buffer Model

To take into account thermal stratification we can extend the previous model with a second water layer [13]. Sinha *et al.* [13] model 2 separate layers, each with their own temperature. T_h and T_c for top (hot) and bottom (cold) layer, respectively. The layer temperatures are monotonically increasing with buffer height, i.e., $T_c \leq T_h$.

Sinha *et al.* argue that a thermocline exists inside the buffer, which limits mixing between top and bottom layer. As a consequence, they assume a hard boundary between both layers. When both layers reach the same temperature, they merge. Both layers diverge when hot water is tapped and, therefore, cold water is added at the bottom. As more cold water is added, the cold (hot) layer's volume increases (decreases) and the boundary between hot and cold layer moves up. We call m_c the cold layer's mass and V_c the cold layer's volume. We add another model variable M indicating if the layers are merged ($M = 1$) or not ($M = 0$). We define the share of cold and hot water as in (3) and (4), respectively.

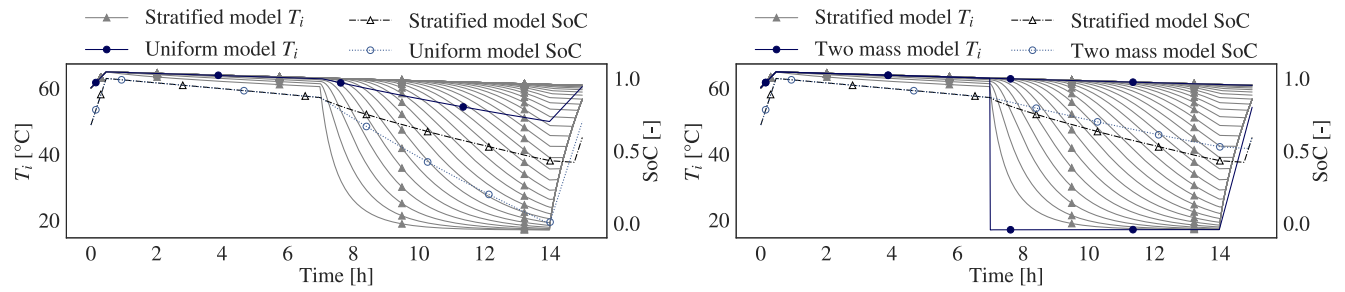
$$X_c = \frac{V_c}{V} \quad (3)$$

$$X_h = \frac{V - V_c}{V} \quad (4)$$

Heat balance is defined by (5) and (6), with $F = 1 - (1 - M)(1 - X_c)$. Thus $F = X_c$ when $M = 0$ and $F = 1$ when $M = 1$.

$$\frac{dT_h}{dt} = \frac{1}{X_h C_{\text{boiler}}}\big(M\dot{Q}_{\text{heat}} + M\dot{m}_w C_{pw}(T_{iw} - T_h) + U(A_s X_h + A_t + MA_t)(T_{\text{amb}} - T_h)\big) \quad (5)$$

$$\frac{dT_c}{dt} = \frac{1}{F C_{\text{boiler}}}\big(\dot{Q}_{\text{heat}} + \dot{m}_w C_{pw}(T_{iw} - T_c) + U((1-M)A_s X_c + MA_s + A_t + MA_t)(T_{\text{amb}} - T_c)\big) \quad (6)$$



(a) Layer temperature(s) and SoC of uniform and stratified buffer model.

(b) Layer temperatures and SoC of two mass and stratified buffer model.

Fig. 1. Comparison between buffer models.

The cold layer's mass and the binary variable M are then determined by (7) and (8), respectively.

$$m_c = \begin{cases} 0 & M = 1 \text{ and } \dot{Q}_{\text{heat}} \neq 0 \\ \int \dot{m}_w dt & \dot{Q}_{\text{heat}} = 0 \end{cases} \tag{7}$$

$$M = \begin{cases} 1 & T_h \leq T_c \text{ or } V \leq V_c \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Heat balance equations (5) and (6) can more easily be interpreted by separating them in three.

1) When the layers are merged and no hot water is tapped both layers have an equal temperature. The model reduces to the uniform case, i.e., (1) with $\dot{m}_w = 0$.
2) When the layers are merged, hot water is tapped but the heating element is on, no thermocline is created. Again, the model reduces to (1), now with $\dot{Q}_{\text{heat}} \neq 0$.
3) Otherwise, the temperature of both layers is described by (5) and (6), with $M = 0$, $F = X_c$ and $m_c = \int \dot{m}_w dt$.

Using this model we define the buffer's SoC as in (9), with $L = \{i \in \{c, h\} | T_i \geq T_{\min}\}$ and $V_i = X_i \cdot V$ for $i \in \{c, h\}$.

$$SoC = \frac{\sum_{i \in L} V_i (T_i - T_{\min})}{V(T_{\max} - T_{\min})}. \tag{9}$$

C. Stratified Buffer Model

We can take the idea of modelling separate layers of water even further. Here, we model $N = 50$ different water layers in the buffer. The stratified buffer model we use has been presented and validated in the lab by Vanthournout *et al.* [14]. This model has been used in DR research several times [1], [16], [17], and for more details we refer to Vanthournout *et al.* [14].

In contrast with the two mass model's variable layer size, here each layer has an equal and constant volume V/N. Furthermore, the temperature T_i of each layer $i \in \{0, \ldots, N\}$ is uniform and monotonically increasing with buffer height ($T_i \leq T_{i+1}$). Vanthournout's model also considers heat exchange between layers.

Equation (9) can also be applied to the stratified model's SoC [14], with $L = \{i \in \{1, \ldots, N\} | T_i \geq T_{\min}\}$ and $V_i = V/N$ for all $i \in \{1, \ldots, N\}$.

D. Comparison

The three models have been simulated according to the same pattern, and Fig. 1 shows their T_i and SoC. Fig 1(a) shows the water temperature of the uniform and stratified buffer on the left axis. At the start, temperature is 60 °C and the heating element is turned on for 40 minutes. Until hour 7, the buffers are left idle, which shows standing heat losses to the environment. The next 7 hours hot water is drawn at 2 ml/s. While taking hot water the stratified model's layers diverge. This highlights the difference between both models. When the heating element is turned on again, the stratified model's layers heat from bottom to top, since the heat source is at the bottom of the tank. The right axis depicts SoC of both models. With no water demand and when all layers of the stratified model have an approximately equal temperature, SoC of both models is almost identical. However, when water is drawn the uniform model's SoC differs from the stratified model. This underestimation of SoC occurs since the uniform model does not capture stratification. While in the stratified model a large portion of layers is still relatively hot, the uniform model assumes the whole buffer's water temperature decreases.

This effect is mitigated by modelling a second layer. Fig. 1(b) shows SoC of the two mass and stratified model match more closely. In the two mass model, as soon as hot water is tapped a second layer is created with temperature $T_c = T_{iw}$. Notice that after hour 7, while the cold layer's temperature is constant, its volume is increasing, resulting in SoC decrease. Fig. 2 visualizes the simplification made by modelling 2 instead of 50 layers. It shows buffer section, i.e., temperature in function of height at different times. Comparing hour 9 and 13 shows the mentioned effect of constant cold layer temperature but raising thermocline, when hot water is being tapped. At hour 14 the heating element is turned on, layers are heated from top to bottom. Due to the two mass model's larger share of water at temperature T_{iw}, this results in a buffer section as shown in the rightmost graph of Fig. 2 at 14h30.

III. Problem Formulation and Algorithms

Demand response of an EWH gives rise to a Sequential Decision Problem (SDP). In earlier work [1], [3], [16], [17] the learning agent's SDP has already been formulated as a Partially Observable Markov Decision Process (POMDP). This section reviews the POMDP and introduces the algorithms used in our work.

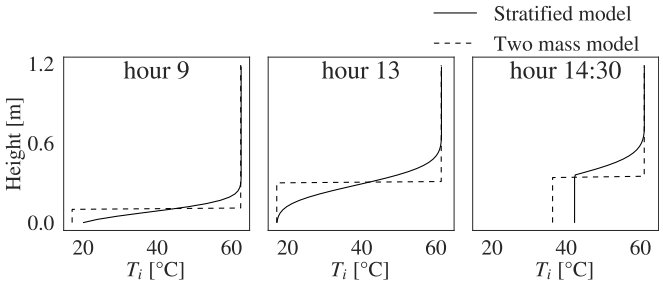


Fig. 2. Section of two mass and stratified buffer model at hour 9, 13 and 14:30.

A. Markov Decision Process

We consider a discrete-time Markov Decision Process (MDP) with time steps of length $\Delta t = 15$ minutes, a one day horizon $T = 96$, state-space $\mathcal{X}$, action-space $\mathcal{U}$, reward-function $r : (\mathcal{X}, \mathcal{U}, \mathcal{X}) \rightarrow \mathbb{R}$ and state-transition probabilities $p(\cdot|x, u)$, modeled by the EWH models. In this MDP, we aim to find a policy $\pi^* : \mathcal{X} \rightarrow \mathcal{U}$ which maximizes total expected discounted rewards, i.e., maximizes $J(\pi)$ with discount factor γ and $r_{t+1} = r(x_t, u_t, x_{t+1})$ [11].

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_{t+1} \right] \tag{10}$$

Due to the difficult system-identification step in residential DR settings $p(\cdot|x, u)$ is not known. Additionally, only part of $\mathcal{X}$ is observable. Therefore, we aim to approximate $\pi^*(x)$ by sampling from $p(\cdot|x, u)$, i.e., by interacting with the EWH. The state-action value function $Q(x, u)$ is estimated using the update rule (11), and (12) defines the policy [18].

$$Q_{i+1}(x_t, u_t) = \mathbb{E}\Big[r_{t+1} + \gamma \max_u Q_i(x_{t+1}, u) | x_t, u_t \Big] \tag{11}$$

$$\pi(x) = \operatorname*{argmax}_u Q(x, u) \tag{12}$$

The next paragraphs give a description of the (observable) state-space, action-space and cost-function.

1) State-Space: Patyn *et al.* [16] investigated the influence of different observable state-space configurations on the performance of a RL agent applied to an EWH. Observing *SoC* results in the best performing control policy. We therefore consider (observable) state x at every time step t to be given by (13), with SoC_t the *SoC* at time-step t. The current quarter of the day t, is transformed to $t_{\cos}$ and $t_{\sin}$ according to (14) and (15).

$$x_t = \big[SoC_{t-3}, SoC_{t-2}, SoC_{t-1}, SoC_t, t_{\cos}, t_{\sin} \big] \tag{13}$$

$$t_{\sin} = \sin(2\pi \cdot t/96) \tag{14}$$

$$t_{\cos} = \cos(2\pi \cdot t/96). \tag{15}$$

2) Action-Space: The MDP has a binary action-space $u \in \mathcal{U} = \{0, 1\}$. Action $u = 1$ implies the EWH's heating element is turned on at rated power, $u = 0$ turns the heating element off. However, the EWH's internal controller maps the agent's action u to a physical action u_{phys} according to (16), with T_b the temperature at the backup sensor location.

$$u_{\text{phys}} = H(u) = \begin{cases} 1 & T_b \leq T_{\min} \\ u & T_b > T_{\min} \text{ and } T_b < T_{\max} \\ 0 & T_b \geq T_{\max} \end{cases} \tag{16}$$

While $H(u)$ is unknown to the agent, it's effect can be observed through the cost-function.

3) Cost-Function: We consider an energy arbitrage DR setting, with Time-of-Use (ToU) pricing. The cost-function defines the MDP's reward as in (17), given energy price λ_t and energy consumption $E_t = H(u_t)P_r\Delta t$.

$$r_{t+1} = -c_{t+1} = -\lambda_t u_{\text{phys},t} P_r \Delta t = -\lambda_t E_t \tag{17}$$

We present the different price profiles used during our experiments in Section V.

B. Double Q-Learning

Double Q-learning (DQL) has been proposed, and later updated, by Van Hasselt *et al.* [19], [20]. It incorporates two NNs: the online network, with parameters θ_t at time step t, and the target network, with parameters θ_t^-. Van Hasselt *et al.* [20] proposes the following update rule for the online network

$$Y_t = r_{t+1} + \gamma Q\left(x_{t+1}, \operatorname*{argmax}_u Q(x_{t+1}, u; \theta_t); \theta_t^- \right) \tag{18}$$

The targets Y_t are calculated using uniform samples from experience replay memory $\mathcal{F}$, containing transitions of the form $\{x_t, u_t, x_{t+1}, r_{t+1}\}$. We use update rule (19) [21], with $\tau = 0.1$, for the target network's parameters.

$$\theta_t^- = \tau \theta_t^- + (1 - \tau)\theta_t \tag{19}$$

All NNs were implemented using the PyTorch framework [22]. They consist of two hidden layers, each with 64 neurons and a ReLU activation function.

C. Fitted Q-Iteration

Due to its proven track record in DR applications we also consider FQI [1], [2], [16] in our experiments. In a discrete-time MDP with a one day horizon ($T = 96$) we can reformulate the SDP as a sequence of T control problems. Approximating the Q-function $\hat{Q}_t(x_t, u_t)$ at each time step t gives rise to T supervised learning problems. Assuming $Q_T = r_{T+1}$, Q-values at time steps $t \in \{0, \ldots, T-1\}$ are defined as in equation (20).

$$\hat{Q}_t(x_t, u_t) = r_{t+1} + \gamma \max_u \hat{Q}_{t+1}(x_{t+1}, u) \tag{20}$$

We calculate $\hat{Q}_t$ using all samples in $\mathcal{F}$ and represent it with a random forest [23]. One of the main advantages of FQI in DR is this division of the SDP into T supervised learning problems. The iterative re-training of all Q-functions inherently allows a changing ToU-price profile, as rewards can be recalculated each iteration, which might be the case in certain DR settings.

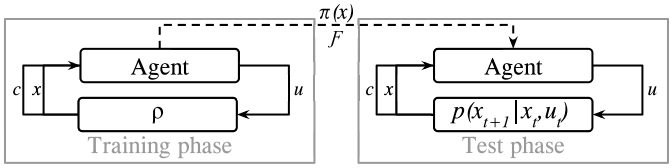


Fig. 3. Policy $\pi(x)$ and experience replay memory $\mathcal{F}$ transfer approach. Training phase model is a distribution ρ over the source domain. The agent observers stratified model dynamics ($p(x_{t+1}|x_t, u_t)$) during test phase.

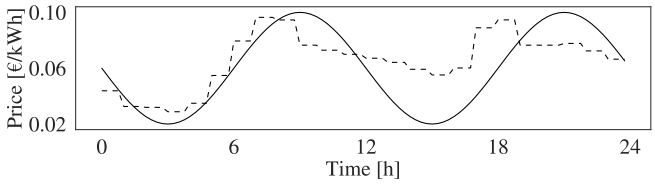


Fig. 4. Sinusoidal price profile (full line) and a one day example of the Belpex price profile (dashed line).

IV. Method

Our objectives are two-fold. We want to keep all benefits of a model-free approach when controlling an EWH in energy arbitrage DR settings, and we want to exploit the limited a priori available information to reduce learning time. The main hypothesis is that pre-training can be used for policy initialization despite the presence of modelling errors, and that it reduces initialization time. In this work, we know the RL agent is going to be applied on an EWH to minimize operating cost when energy consumption is charged with a (given) ToU price. The stratified model presented in Section II-C, and with parameters as given in Table I, represents the target domain. This implies we can only use it as a virtual test-bed. The uniform and two mass model represent the source domain, i.e., simplified (or approximated) models of the target domain. Our objectives then translate to training policies that can control the EWH under the stratified model's dynamics $p(x_{t+1}|x_t, u_t)$, while reducing learning time using its approximate dynamics $\hat{p}(x_{t+1}|x_t, u_t)$.

All experiments consist of a training and test phase, as illustrated in Fig. 3. During training, the agent samples from $\hat{p}(x_{t+1}|x_t, u_t)$, by interacting with the source domain. It saves the samples in $\mathcal{F}$ and updates $\pi(x)$ using either DQL or FQI. To account for discrepancies between source and target domain, i.e., modelling differences, we include variability in the former [11]. We use domain randomization and expose the agent to a distribution ρ over environments $\hat{p}(x_{t+1}|x_t, u_t) \sim \rho$ [10]. Tobin *et al.* [10] state that with enough variability during the training phase, the RL agent will be able to generalize to dynamics seen in the test phase. Because, instead of training a policy that can perform the EWH control task under one dynamics model, we train a policy that can perform the task over a variety of models.

In the remainder of this work, we assume the EWH's height h, diameter d and rated power P_r are known in the training phase and equal to the value in Table I. Furthermore, the target domain's thermal transmittance U equals $0.75\,\mathrm{W/(m^2K)}$. In the training phase, we assume a normal distribution over $U \sim \mathcal{N}(\mu, \sigma^2)$ in the source domain, with the value of μ and σ depending on the experiment. During this phase, the agent's objective is then to maximize expected return across a distribution of dynamics models ρ [11]. The agent learns a control policy using DQL or FQI. The training phase lasts for $T_{\text{train}} = 15T$ simulation days ($t = 1440$). We assume a separate tap water profile $\dot{m}_w$ of 15 days is available for this phase. Both in the training and test phase we use an ϵ-greedy exploration policy [18]. After pre-training, the policy π and/or the replay memory $\mathcal{F}$ are transferred, as shown in Fig. 3. The transfer approach differs slightly for DQL and FQI.

A. DQL

The parameters θ of the two NNs are iteratively updated during training phase and encapsulate information about the MDP. The target domain uses the final source domain's θ and θ^- as initial values. The experience replay memory's maximum size $|\mathcal{F}|_{\max}$ is 35 days, or $|\mathcal{F}|_{\max} = 35 \cdot 96$ state-transitions. Each time step t, we perform one update of online and target NN according to (18) and (19), with a batch of size $3T = 288$ uniformly sampled from $\mathcal{F}$.

B. FQI

At each iteration the whole set of random forests is refitted, rather than updated. Therefore, we only transfer the replay memory $\mathcal{F}$ from source domain to target domain. This is an extension of the MAFQI [8], [9] idea. Since, while Lampe and Riedmiller [8] propose to train a NN to approximate the state-transition function and add virtual tuples to $\mathcal{F}$, we use a physics-based approximate model with domain randomization to (initially) populate $\mathcal{F}$. Again, $|\mathcal{F}|_{\max} = 35T$. This is sufficiently large, as Costanzo *et al.* [9] show FQI converges after approximately 20 days. As a result of this limitation, source domain state-transitions start to be removed from $\mathcal{F}$ after 20 simulated test days ($|\mathcal{F}|_{\max} - T_{\text{train}} = 20T$).

V. Experiments and Results

The experiments aim to verify our main hypothesis: is it faster to fine-tune a DR controller learned on an approximate model rather than learn one from scratch? All simulation setups and their results are presented next. The same experiment has been run 20 times, in order to account for the variability present in the training and test phase. We drew new samples $U \sim \mathcal{N}(\mu, \sigma^2)$ each of the 20 times. The two ToU price profiles divide the experiments into two main categories. First, we present the experiments which use a sinusoidal price profile. Thereafter, we display the ones using the Belgian day-ahead wholesale electricity prices (of 2018) [24].

A. Sinusoidal Price Profile

Fig. 4 shows the sinusoidal price profile. We have based this tariff both on the Belgian day-ahead prices and on the ToU price profile which has been proposed to one of the Belgian energy regulators (VREG) [25]. Prices vary throughout the day, but the same profile occurs every day.

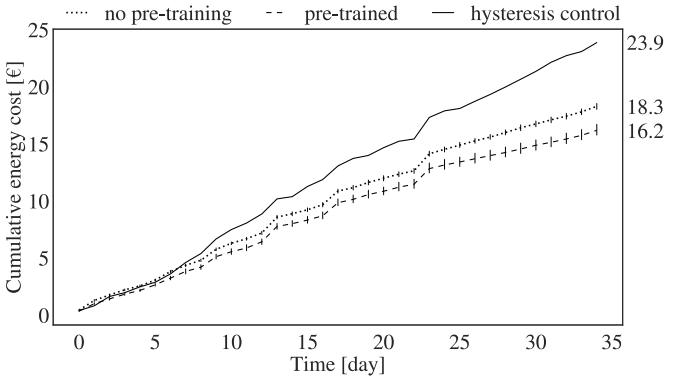


Fig. 5. Cumulative cost comparison: DQL-sin experiment. Mean of 20 simulations, vertical bars indicate 95 % confidence interval.

TABLE III
TOTAL COST AND P-VALUES FOR INTERVALS OF 7 DAYS: DQL-SIN EXPERIMENT

Days	[0,6]	[7,13]	[14,20]	[21,27]	[28,34]	**[0,34]**
p-value	$2e^{-6}$	0.029	0.007	$2.1e^{-5}$	$4.9e^{-4}$	$\mathbf{3e^{-9}}$

1) Double Q-Learning: Direct vs. Pre-Training: In a first experiment we have applied DQL directly in the target domain, and compare it with the pre-training approach as presented in Section IV-A. We employed the two mass model, presented in Section II-B, in the source domain, as this resembles the target domain (stratified model) most closely. Each simulation run we sampled five two mass models, with $U \sim \mathcal{N}(0.75, 0.1^2)$.

Fig. 5 shows the cumulative cost of three control approaches over 35 days. The most expensive control approach, with a final cost of €23.9, is the hysteresis controller. This controller acts according to (16), with $u = 0$. The dotted and dashed line show the cumulative cost of the direct and pre-training approach. The bars depict the 95 % confidence bound (over 20 simulation runs). As known from previous research [1], [16], [17], RL (without pre-training) manages to reduce cost compared to a hysteresis controller. Here, cost has been reduced by 23.4 %. With DQL and given this price profile, pre-training reduces cost 8.8 % further, and has a total cost reduction of 32.2 % compared to the hysteresis controller.

Fig. 6 compares performance of both approaches. It separates the simulation days in 5 intervals of 7 days and shows the mean total cost and standard deviation (of 20 simulation runs). Table III shows the t-test's p-value for the mean of these costs, i.e., the probability that these two distributions have an identical average. The table shows total cost of all intervals differs significantly for this experiment. Additionally, the whole simulation run's total cost differs significantly, as the p-value is close to zero. While cost between days within one approach can vary because of tap water demand, the same days have the same tap demand across all approaches.

Fig. 7 shows the policy in the target domain for both DQL approaches. The top row visualizes the policy learned from scratch, after 5 and 35 days. The bottom row shows the same after pre-training. To create this figure, snapshots of the target domain policy NN were saved during simulation at the given time steps. Afterwards, random state-space samples were fed

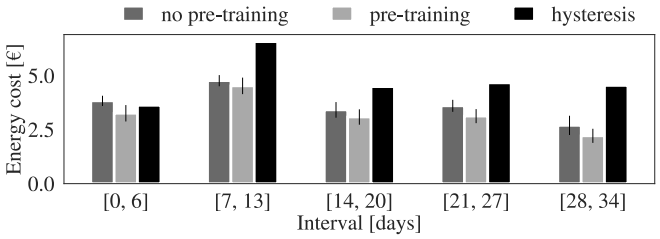


Fig. 6. Cost comparison of intervals of 7 days: DQL-sin experiment.

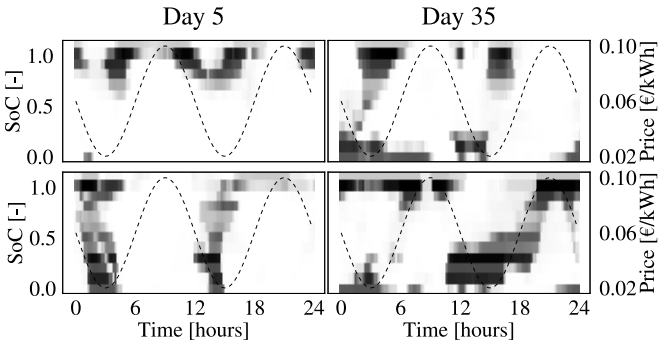


Fig. 7. Policy comparison: DQL-sin experiment. Top row: policy without transfer learning, bottom row: policy with transfer learning. Left column: snapchot of policy after 5 days, right column: snapchot of policy after 35 days. Back indicates $u_t = \pi(x_t) = 1$.

through these snapshots. As is clear from (13), each state x consists of four *SoC* values. Thus, given a certain value for SoC_t, the policy outcome can be either $u = \pi(x) = 0$ or $u = \pi(x) = 1$, depending on the other three *SoC* values. Fig. 7 is a two-dimensional representation of the policy, i.e., states with the same value for SoC_t and t are depicted on the same point in this two-dimensional space, although they might have different values for SoC_{t-1}, SoC_{t-2}, SoC_{t-3}. Black indicates the EWH is turned on ($\pi(x) = 1$) for that particular time-step t and SoC_t, for all state-space samples. Lighter shades indicate energy consumption only for certain samples with the same SoC_t and t (but possibly different values for SoC_{t-1}, SoC_{t-2}, SoC_{t-3}).

The top left figure shows that only a limited amount of the state-space has been explored in the direct approach, and the policy is initialized only in this part. After pre-training, the policy is already initialized over a larger part of $\mathcal{X}$. After 5 days, the pre-trained agent has learned a basic policy, turning the EWH on when prices are low. After 35 days, the RL agent that started from scratch has also learned this behaviour. Additionally, when *SoC* turns out to be low before the highest peak, the agent turns the EWH on. We see the policy learned after 35 days and with pre-training is similar, but the big charging cycle has moved to the afternoon price drop. This results in a cheaper policy, as most of the DHW consumption occurs in the evening [15]. In the morning it is sufficient to charge to ±50 %.

2) Fitted Q-Iteration: Direct vs. Pre-Training: The same figures are shown, now for the FQI approach. Fig. 8 shows the cumulative cost of the compared control approaches. Again, the hysteresis controller is the most expensive, while the pre-training approach is the cheapest. Table IV shows that the

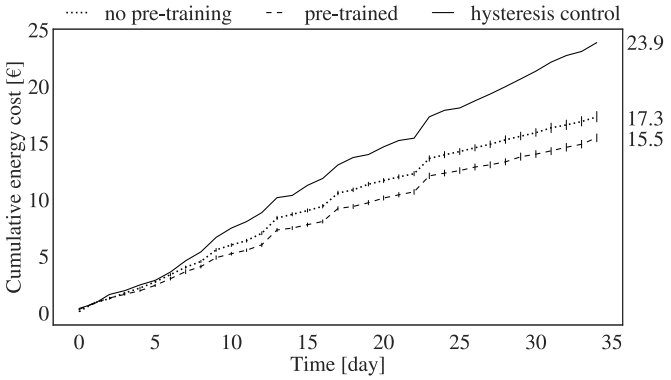


Fig. 8. Cumulative cost comparison FQI-sin experiment. Mean of 20 simulations, vertical bars indicate 95% confidence interval.

TABLE IV
TOTAL COST AND P-VALUES FOR INTERVALS OF 7 DAYS: FQI-SIN EXPERIMENT

Days	[0,6]	[7,13]	[14,20]	[21,27]	[28,34]	**[0,34]**
p-value	$3e^{-4}$	$6.3e^{-8}$	$2.5e^{-5}$	0.04	0.88	$\mathbf{4e^{-7}}$

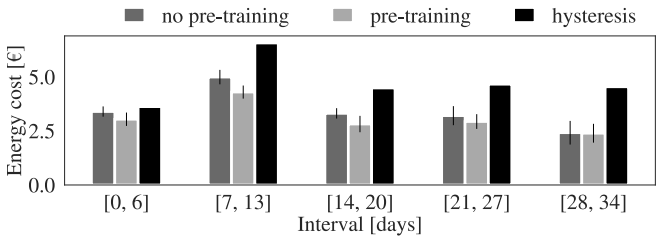


Fig. 9. Cost comparison of intervals of 7 days: FQI-sin experiment.

difference between pre-training or not is significant for all but the last interval. Despite learning from scratch reducing total cost with 27.6 %, compared to hysteresis control, pre-training reduces cost with 35.1 %. Fig. 9 compares all three controllers in this experiment. It shows both FQI-based RL approaches are already cheaper than the hysteresis controller in the first interval. The figure also shows pre-training results in additional cost gains, starting from the first interval.

3) Comparison With Model Predictive Control: Of the three presented buffer models, the uniform model is the only one that is linear. For this case, the source domain's control problem can thus be formulated as a Mixed Integer Linear Problem (MILP), given in (21). Therefore, it is possible to use Model Predictive Control (MPC). This experiment compares MPC with both transfer learning approaches, when using the uniform model. With MPC, the target domain's (observable) state is sampled at time step t. Here, the target domain's (stratified buffer) backup sensor temperature at that time step ($T_{b,t}$) is used as initial state in the source domain (uniform model), as shown in (21e), and $U = 0.75\,\mathrm{W/(m^2K)}$ in both domains. The MILP's solution is a vector of length T with the source domain's optimal control actions for period $[t, t+T]$. After applying action u_t in the target domain, the procedure is repeated, and this for every time step t. For a more elaborate explanation of MPC we refer to other literature [26]. We solve the MILP

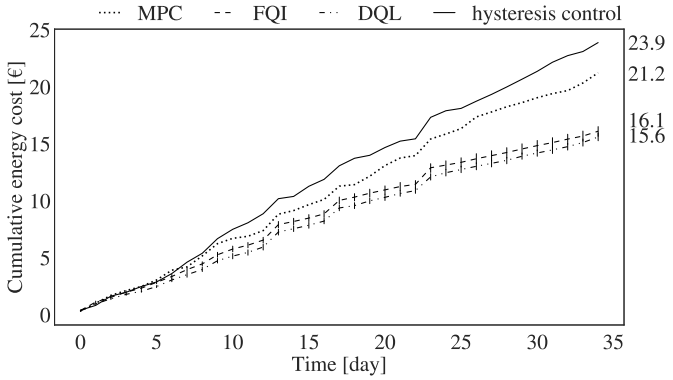


Fig. 10. Cumulative cost comparison: FQI and DQL vs. MPC experiment (uniform model). Mean of 20 simulations, vertical bars indicate 95 % confidence interval.

TABLE V
RESULTS FOR DIFFERENT DISTRIBUTION PARAMETERS OVER U

U		$\mathcal{N}(0.75, 0.1^2)$	$\mathcal{N}(0.55, 0.1^2)$
FQI [€]	μ	15.49	15.25
	σ	0.79	0.97
DQL [€]	μ	16.20	16.08
	σ	1.04	1.02

using the Gurobi solver [27].

$$\min_{u_k} \quad \sum_{k=t}^{t+T} \lambda_k u_k P_r \Delta t \tag{21a}$$

$$\text{subject to} \quad (1) \tag{21b}$$

$$T_{L_k} \geq T_{\min} \tag{21c}$$

$$T_{L_k} \leq T_{\max} \tag{21d}$$

$$T_{L_t} = T_{b,t} \tag{21e}$$

Fig. 10 shows the cumulative cost of all 4 controllers (MPC, pre-trained FQI, pre-trained DQL and hysteresis control). Pre-trained DQL's final cost is € 15.6, and thus 26 % cheaper than MPC, with a final cost of € 21.2. Pre-training FQI with the uniform model results in a total cost of € 16.1. It is thus clear it does matter to capture the non-linearities of the stratified model, which a RL agent is able to do. Although RL manages to capture relevant information from the uniform model, the model does not seem to be sufficiently accurate for MPC. Furthermore, this figure also shows there is barely any performance difference between FQI and DQL.

4) U Out of Distribution: In a last experiment with this price profile, we investigated the effect of a bad distribution over U, both for DQL and FQI. Apart from sampling U from $\mathcal{N}(0.55, 0.1^2)$ the procedure is equal to the first and second experiment. The target domain's thermal transmittance U is now 2σ away from the source domain's mean value.

Table V shows the results. For both algorithms, the performance with a *good* guess for the distribution over U (target domain value equal to source domain mean) and with a *bad* guess is very similar. This indicates that pre-trained policies are indeed able to generalize over different buffer model parameters, and that we can use RL to further fine-tune policies in the target domain.

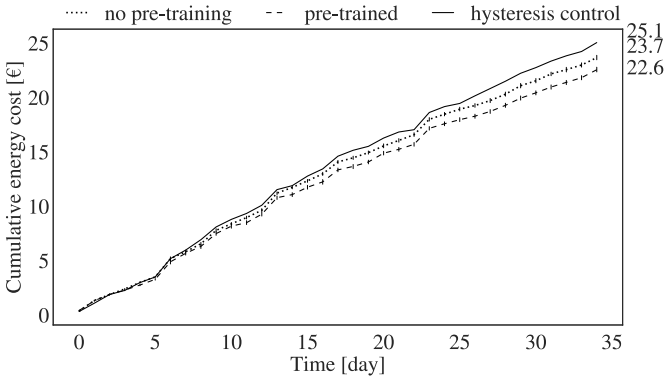


Fig. 11. Cumulative cost comparison: FQI-Belpex experiment. Mean of 20 simulations, vertical bars indicate 95 % confidence interval.

B. Belpex Price

In a final experiment, we have used the Belgian day-ahead electricity prices [24] (dashed line in Fig. 4), and compared performance difference between the use of pre-training or not. Belpex is an hourly-varying price profile which differs every day. E.g., Fig. 4 shows the first day of the test-set prices. FQI separates the SDP in T supervised learning problems and refits the regressor for every control step t. Ruelens *et al.* [1] propose to exploit this property when dealing with a time-varying ToU price profile, which is different every day, by recalculating all rewards in $\mathcal{F}$ at every step. As DQL is an online learning algorithm, adapting it in a similar way is non-trivial, as information of previous rewards is encapsulated in the NN's weights. Another possibility, as explored by Cao *et al.* [28], is to add future prices to the state. Thus, for the FQI-based controller no additional adaptations are needed to the state, but every training day the rewards in $\mathcal{F}$ need to be recalculated. For the DQL-based controller on the other end, this recalculation step is unnecessary, but the observed state at time step t is now given by (22).

$$x_t = \left[SoC_{t-3}, \ldots, SoC_t, t_{\cos}, t_{\sin}, \lambda_t, \ldots, \lambda_{t+24}\right] \quad (22)$$

First all FQI results are presented. Fig. 11 shows the cumulative cost over the simulation period for all considered FQI controllers. While their order stays constant, the difference between them reduces. With this price profile, direct RL is 5.5 % cheaper than the hysteresis controller and the transfer learning approach is another 4.4 % cheaper. Compared to the same experiment with the sinusoidal price profile, the cost gains are smaller.

This can be explained by the smaller valleys and peaks in the price profile, as is clear from Fig. 12. This figure shows the target domain's final five simulation days, for the pre-trained case. The top graph depicts Belpex price on the right axis and the buffer's *SoC* on the left axis. The grey areas indicate if the EWH is consuming power ($u_{\text{phys}} = 1$) or not ($u_{\text{phys}} = 0$). The bottom graph depicts DHW consumption. The control policy turns the EWH mostly on when prices are relatively low. However, completely avoiding energy consumption during higher priced hours is not always possible due to hot water consumption.

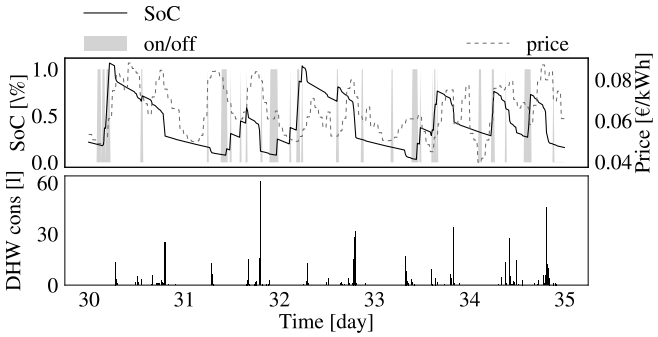


Fig. 12. Final five simulation days: FQI-Belpex experiment.

TABLE VI
TOTAL COST AND P-VALUES FOR INTERVALS OF 5 DAYS: FQI-BELPEX EXPERIMENT. MEAN OF 20 SIMULATIONS, VERTICAL BARS INDICATE 95% CONFIDENCE INTERVAL

Days		[0,6]	[7,13]	[14,20]	[21,27]	[28,34]	**[0,34]**
No pre-training	μ	5.19	6.02	4.34	4.17	3.95	**23.67**
$\sum \lambda_t E_t$ [€]	σ	0.40	0.16	0.25	0.32	0.19	**0.46**
Pre-trained	μ	4.90	5.90	4.09	3.84	3.84	**22.56**
$\sum \lambda_t E_t$ [€]	σ	0.45	0.18	0.15	0.22	0.22	**0.49**
p-value		0.04	0.03	$7e^{-4}$	$6e^{-4}$	0.11	$\mathbf{1.2e^{-8}}$

Table VI divides the simulation period in 5 intervals, each of 7 days. The table shows that each group's mean cost for direct FQI is larger than the group's cost with pre-trained FQI. The difference is, again, significant for all but the last group. As expected since $\mathcal{F}$ is almost only filled with target domain transitions at that time.

With pre-trained DQL and, using states as defined in (22), the Belpex price profile resulted in a mean total cost of € 23.22 with a standard deviation of 0.64. These results seem to indicate FQI is more suited in a setting with daily varying prices. Additionally, with the Belpex price profile, MPC resulted in a total cost of € 25.5. This result indicates that, with this price profile, more system-identification effort is needed to get satisfactory MPC performance.

VI. CONCLUSION AND FUTURE WORK

We demonstrated the use of pre-training with domain randomization in a residential DR setting for two different RL algorithms and with two ToU price profiles. We show adapting a pre-trained policy to the target domain is significantly faster than starting from scratch. Pre-trained policies allow for cost savings immediately at the start of operation, which, in the considered cases, results in a significant price reduction throughout the simulated period. Using DQL, pre-training results in 8.8 % cost reduction compared to starting from scratch and a 32.2 % reduction compared to a hysteresis controller. Although the pre-training approach differs slightly between DQL and FQI, both algorithms benefit very similarly from it. Moreover, despite the two mass model's more accurate *SoC* estimate, compared to the uniform model, the RL agent manages to benefit from pre-training almost equally. Average total cost with the two mass model is € 16.2 while it is € 15.6 with the uniform model, for the DQL agent. Our experiments also showed pre-trained RL agents, with domain

randomization, show similar cost savings when there is a larger discrepancy between source domain and target domain. And, this discrepancy can be caused by modelling (uniform model) or by system-identification (bad guess of model parameter U).

Finally, we showed it is relevant to use a non-linear policy which adapts to the target domain, as this results in 26 % cheaper operation than applying optimal source domain control actions in the target domain, i.e., MPC. While state-of-the-art MPC might result in better performance, our results indicate that the uniform model suffices for RL but does not suffice for MPC.

It is not certain that dissimilarity between uniform / two mass model and stratified model is as large as the dissimilarity between the stratified model and a real buffer. Therefore, future work is directed towards verifying the presented approach for transfer from simulation to practice. Additionally, we aim to investigate other methods of including prior knowledge in the policy as we have showed this can result in better policies.

REFERENCES

[1] F. Ruelens, B. J. Claessens, S. Quaiyum, B. De Schutter, R. Babuška, and R. Belmans, "Reinforcement learning applied to an electric water heater: From theory to practice," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3792–3800, Jul. 2018.

[2] M. Liu, S. Peeters, D. S. Callaway, and B. J. Claessens, "Trajectory tracking with an aggregation of domestic hot water heaters: Combining model-based and model-free control in a commercial deployment," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5686–5695, Sep. 2019.

[3] T. Peirelinck, F. Ruelens, and G. Deconinck, "Using reinforcement learning for optimizing heat pump control in a building model in modelica," in *Proc. IEEE Int. Energy Conf. (ENERGYCON)*, Limassol, Cyprus, 2018, pp. 1–6.

[4] C. Patyn, F. Ruelens, and G. Deconinck, "Comparing neural architectures for demand response through model-free reinforcement learning for heat pump control," in *Proc. IEEE Int. Energy Conf. (ENERGYCON)*, Limassol, Cyprus, Jun. 2018, pp. 1–6.

[5] O. De Somer, A. Soares, T. Kuijpers, K. Vossen, K. Vanthournout, and F. Spiessens, "Using reinforcement learning for demand response of domestic hot water buffers: A real-life demonstration," in *Proc. IEEE PES Innovat. Smart Grid Technol. Conf. Eur. (ISGT-Europe)*, Torino, Italy, 2017, pp. 1–7.

[6] H. Kazmi, F. Mehmood, S. Lodeweyckx, and J. Driesen, "Gigawatt-hour scale savings on a budget of zero: Deep reinforcement learning based optimal control of hot water systems," *Energy*, vol. 144, pp. 159–168, Feb. 2018.

[7] F. Ruelens, B. J. Claessens, S. Vandael, B. De Schutter, R. Babuška, and R. Belmans, "Residential demand response of thermostatically controlled loads using batch reinforcement learning," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2149–2159, Sep. 2017.

[8] T. Lampe and M. Riedmiller, "Approximate model-assisted neural fitted Q-iteration," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Beijing, China, Sep. 2014, pp. 2698–2704.

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

[10] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, Dec. 2017, pp. 23–30.

[11] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, QLD, Australia, 2018, pp. 3803–3810.

[12] A. A. Farooq, A. Afram, N. Schulz, and F. Janabi-Sharif, "Grey-box modeling of a low pressure electric boiler for domestic hot water system," *Appl. Thermal Eng.*, vol. 84, pp. 257–267, Jun. 2015.

[13] R. Sinha, B. B. Jensen, J. R. Pillai, C. Bojesen, and B. Moller-Jensen, "Modelling of hot water storage tank for electric grid integration and demand response control," in *Proc. 52nd Int. Univ. Power Eng. Conf. (UPEC)*, Heraklion, Greece, 2017, pp. 1–6.

[14] K. Vanthournout, R. D'hulst, D. Geysen, and G. Jacobs, "A smart domestic hot water buffer," *IEEE Trans. Smart Grid*, vol. 3, no. 4, pp. 2121–2127, Dec. 2012.

[15] S. Edwards, I. Beausoleil-Morrison, and A. Laperrière, "Representative hot water draw profiles at high temporal resolution for simulating the performance of solar thermal systems," *Solar Energy*, vol. 111, pp. 43–52, Jan. 2015.

[16] C. Patyn, T. Peirelinck, and G. Deconinck, "Intelligent electric water heater control with varying state information," in *Proc. IEEE Int. Conf. Commun. Control Comput. Technol. Smart Grids (SmartGridComm)*, Aalborg, Denmark, 2018, pp. 1–6.

[17] T. Peirelinck, F. Spiessens, C. Hermans, and G. Deconinck, "Double Q-learning for demand response of an electric water heater," in *Proc. IEEE PES Innovat. Smart Grid Technol. Eur. (ISGT Europe)*, Bucharest, Romania, 2019, p. 5.

[18] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2017.

[19] H. Van Hasselt, "Double Q-learning," in *Advances in Neural Information Processing Systems 23*. Red Hook, NY, USA: Curran Assoc., Inc., 2010, pp. 2613–2621.

[20] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-Learning," in *Proc. 30th AAAI Conf. Artif. Intell. (AAAI)*, 2016, pp. 2094–2100.

[21] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. 4th Int. Conf. Learn. Represent.*, 2016, pp. 1–14.

[22] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach and H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Assoc., Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[23] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[24] *EPEX SPOT Belgium*, EPEX SPOT Brussels, Brussels, Belgium, 2018. [Online]. Available: https://www.belpex.be/

[25] R. D'hulst, A. Delnooz, E. Laes, and D. Six, "Onderzoek naar de tariefstructuur van de periodieke distributienettarieven Finaal rapport," Energie wijzer, Flanders, Belgium, Rep., 2018. [Online]. Available: https://www.vreg.be/sites/default/files/20180111_studie_vreg_statusrapport_v11_-_eindrapport.pdf

[26] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Santa Barbara, CA, USA: Nob Hill Publ., LLC, 2019.

[27] *Gurobi Optimizer Reference Manual*, Gurobi Optim., LLC, Houston, TX, USA, 2020. [Online]. Available: http://www.gurobi.com

[28] J. Cao, D. Harrold, Z. Fan, T. Morstyn, D. Healey, and K. Li, "Deep reinforcement learning-based energy storage arbitrage with accurate lithium-ion battery degradation model," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4513–4521, Sep. 2020.

Thijs Peirelinck (Graduate Student Member, IEEE) received the B.Sc. degree from KU Leuven, Belgium, in 2014, the master's degree in energy engineering from KTH Stockholm, Sweden, and INP Grenoble, France, and the master's degree (magna cum laude) in artificial intelligence from KU Leuven in 2017, where he is currently pursuing the Ph.D. degree with the Electrical Energy and Computer Architectures (ELECTA) Research Group. His main research interest is residential demand response.

Chris Hermans received the master's and Ph.D. degrees in computer science from Hasselt University, specializing in the area of Visual Computing 2005 and 2011, respectively. In 2016, he worked with 4DDynamics, where he developed automated 3-D scanning solutions for the consumer market. Afterwards, he joined VITO, the Flemish Institute for Technological Research, where he is performing research in the domain of smart grid related algorithms, modeling and optimisation (AMO). His interests include computer vision, image processing, machine learning, distributed computing, and generally anything involving applied mathematics.

Fred Spiessens received a degree in electronics 1988, the master's degree in computer science from VUB 1992, and the Ph.D. degree in software security from UCL Louvain-la-Neuve in 2007, while he was working fulltime as a Plant Operator at an oil refinery. He graduated with "Great Distinction" in both. He has more than 30 years of mixed industry-academic experience. After having worked for Sony on several innovative energy projects in collaboration with VITO, he joined VITO's Energy team as a Senior Researcher in 2014. As a researcher, he combines techniques from Mathematical Optimisation, Machine Learning and Constraint Programming into robust and scalable algorithms for renewable energy management. He also co-supervises/co-supervisor of Ph.D. students in collaboration with Flemish Universities.

Geert Deconinck (Senior Member, IEEE) received the M.Sc. degree in electrical engineering and the Ph.D. degree in engineering science from KU Leuven, Belgium, in 1991 and 1996, respectively. He is currently a Full Professor with KU Leuven, where he heads the research group on Electrical Engineering Systems and Applications and the EnergyVille Research Center. His research interests include robust distributed coordination and control, specifically in the context of smart grids.