A hybrid policy gradient and rule-based control framework for electric vehicle charging

Brida V. Mbuwir[a,b], Lennert Vanmunster[b], Klaas Thoelen[a,b,*], Geert Deconinck[a,b]

[a]*EnergyVille, Thor Park 8310, 3600 Genk, Belgium*
[b]*ESAT-Electa, KU Leuven, Kasteelpark Arenberg 10 bus 2445, 3001 Leuven, Belgium*

Abstract

Recent years have seen a significant increase in the adoption of electric vehicles, and investments in electric vehicle charging infrastructure and rooftop photo-voltaic installations. The ability to delay electric vehicle charging provides inherent flexibility that can be used to compensate for the intermittency of photo-voltaic generation and optimize against fluctuating electricity prices. Exploiting this flexibility, however, requires smart control algorithms capable of handling uncertainties from photo-voltaic generation, electric vehicle energy demand and user's behaviour. This paper proposes a control framework combining the advantages of reinforcement learning and rule-based control to coordinate the charging of a fleet of electric vehicles in an office building. The control objective is to maximize self-consumption of locally generated electricity and consequently, minimize the electricity cost of electric vehicle charging. The performance of the proposed framework is evaluated on a real-world data set from EnergyVille, a Belgian research institute. Simulation results show that the proposed control framework achieves a 62.5% electricity cost reduction compared to a business-as-usual or passive charging strategy. In addition, only a 5% performance gap is achieved in comparison to a theoretical near-optimal strategy that assumes perfect knowledge on the required energy and user behaviour of each electric vehicle.

Keywords: electric vehicles, smart charging, proximal policy optimization, reinforcement learning

*Corresponding author
Email address: `klaas.thoelen@kuleuven.be` (Klaas Thoelen)

1 List of symbols

$\mathcal{A}$	Action space
a_t	Action at time step t
${}^{i}C_t^{anx}$	Anxiety cost of EV_i at time step t
$C_{t,m}^{cons}$	Consumption cost in minute m during time step t
$C_{t,m}^{elec}$	Electricity cost (consumption + injection) in minute m of t
C^{inj}	Electricity injection cost
$\mathbb{E}$	Expected value
${}^{i}E_t^{ch}$	Energy used to charge EV_i during time step t
${}^{i}E_t^{ch,ses}$	Total energy charged in the session of EV_i up to time step t
${}^{i}E_t^{rem}$	Remaining energy [kWh] for EV_i in the RTC algorithm, i.e. the energy that still needs to be charged before the end of time step t
${}^{i}E_{t,m}^{req}$	Required energy [kWh], i.e. energy that still needs to be charged before departure, of EV_i in minute m of time step t
G	Expected (discounted) return
K_1, K_2, K_3	Coefficients in anxiety cost function ${}^{i}C_t^{anx}$
2 L_c	Length of a control time step [minutes]
L^{CLIP}	Clipped loss function in PPO
L^{CLIP2}	Augmented, clipped loss function in PPO
L^{ORIG}	Unconstrained loss function in PPO
m	Minute index in a control slot
${}^{i}M_t^{end}$	Index of last minute of EV_i in time step t
${}^{i}M_t^{start}$	Index of first minute of EV_i in time step t
N_{ev}	Fleet size, i.e. number of EVs controlled by the agent
N_{fut}	Number of future time steps in state vectors for PV and electricity price forecasts
N_{par}	Number of power partitions in the heuristic dispatch algorithm in the aggregate MDP
N_{past}	Number of past time steps in state vectors of the PV forecast
P_t^a	Aggregate fleet charging power [kW] action (denormalised output of the actor network in the aggregate MDP) at time step t
${}^{i}P_t$	Charging power [kW] action (denormalised output of the actor network in the base and hidden MDPs) of EV_i at time step t

	${}^{i}P_t^{b}$	Clipped charging power [kW] (output of the backup controller or heuristic dispatch) of EV$_i$ at time step t
	${}^{i}P_t^{b,max}$	Maximum charging power [kW] for backup controller or heuristic dispatch of EV$_i$ at time step t
	${}^{i}P_t^{b,min}$	Minimum charging power [kW] for backup controller or heuristic dispatch of EV$_i$ at time step t
	P^{cons}	Historical grid power consumption [kW]
	P^{inj}	Historical grid power injection [kW]
	${}^{i}P^{lim}$	Absolute maximum charging power [kW] of EV$_i$
	$\boldsymbol{P}_t^{fpv}$	State vector at time t of PV power forecast [kW]
	$\boldsymbol{P}_t^{fpv,fut}$	State vector at time t of future PV power forecasts [kW] (after t)
	$\boldsymbol{P}_t^{fpv,past}$	State vector at time t of past PV power forecasts [kW] (before t)
	$P_t^{fpv,rest}$	State parameter containing the average PV power forecast [kW] of the rest of the day after $\boldsymbol{P}_t^{fpv,fut}$
3	$P_m^{max,\Sigma}$	Total maximum charging power [kW] in the RTC algorithm for minute m
	$P_m^{min,\Sigma}$	Total minimum charging power [kW] in the RTC algorithm for minute m
	P_t^{pv}	PV power generation [kW] at time t, minute 0 ($P_{t,m=0}^{pv}$)
	$\boldsymbol{P}_t^{pv}$	State vector of PV power generation [kW] at time step t
	$P_{t,m}^{pv}$	PV power generation [kW] in minute m of time step t
	$P^{pv,net}$	Historic net PV power generation [kW] available for charging the EV fleet
	$\boldsymbol{P}_t^{pv,past}$	State vector of past PV power generation [kW] at time step t
	$P^{pv,scaled}$	Scaled PV power generation [kW], $P^{pv,scaled} \equiv 0.2 \times P^{pv,tot}$
	$P^{pv,tot}$	Historic PV power generation [kW]
	${}^{i}\boldsymbol{P}_t^{r}$	Vector with the charging powers [kW] of EV$_i$ in each minute m of time step t

${}^{i}P^{r}_{t,m}$	Charging power [kW] of EV$_i$ in minute m of time step t
${}^{i}P^{r,max}_{m}$	Maximum charging power [kW] in the RTC algorithm for EV$_i$ in minute m in time step t
${}^{i}P^{r,min}_{m}$	Minimum charging power [kW] in the RTC algorithm for EV$_i$ in minute m in time step t
pr	Probability ratio in PPO algorithm
Q	Action-value function
ρ	Reward function
$\mathcal{S}$	State space
s_t	State at time t
t	Time step index in an episode
T	Final time step or number of time steps in an episodic environment
f	Transition function
${}^{i}T^{arr}$	Arrival time of EV$_i$ (unit = time slot)
4 ${}^{i}T^{dep}$	Departure time of EV$_i$ (unit = time slot)
V	State-value function
${}^{i}\boldsymbol{X}_t$	State vector of EV$_i$ at time step t
$\boldsymbol{Z}_t$	Vector containing the aggregated fleet state parameters at time step t
α	Learning rate
γ	Discount factor
$\Delta{}^{i}T^{dep}$	Time left until departure for EV$_i$ at time t (unit = time steps)
θ	Function approximator parameters (e.g. neural network weights)
κ	Flexibility factor in the RTC algorithm
λ^{Belpex}	Belpex day-ahead electricity price [€/kWh]
$\bar{\lambda}^{cons}$	Average TOU grid consumption price in the current day (between 7:00 and 20:00)
$\boldsymbol{\lambda}^{cons}_{t}$	State vector of TOU grid consumption price [€/kWh] at time step t

$\lambda_{t,m}^{cons}$	TOU grid consumption price [€/kWh] in minute m of time step t
$\boldsymbol{\lambda}_t^{cons,fut}$	State vector at time step t of future TOU grid consumption prices [€/kWh] (after t)
$\lambda_t^{cons,rest}$	State parameter containing the average TOU grid consumption prices [€/kWh] of the rest of the day after $\boldsymbol{\lambda}_t^{cons,fut}$
λ^{inj}	Electricity injection price [€/kWh]
Λ	Advantage function
π	Policy function
${}^{i}\tau$	Flexibility of EV_i in the heuristic dispatch algorithm of $\mathrm{MDP}_{\mathrm{agg}}$
∇	Nabla-operator

1. Introduction

The increasing concern on the effects of greenhouse gas emissions has led to an increase in the use of renewable energy sources (RESs) and the electrification of transport. While the decreasing cost of photo-voltaic (PV) installations has led to an increase in the number of buildings with rooftop PV installations, the electrification of transport has led to several incentive programs to encourage the use of electric vehicles (EVs). For example, the EV30@30 campaign has set a target of at least 30% market share of EVs in the Electric Vehicle Initiative member states by 2030 [1]. The increase in the number of EVs, however, significantly alters the electricity demand curve [2]. A typical example of an alteration of the electricity demand curve can be seen in office buildings with EV charging infrastructure where EVs tend to arrive at the same time, see Fig. 1.

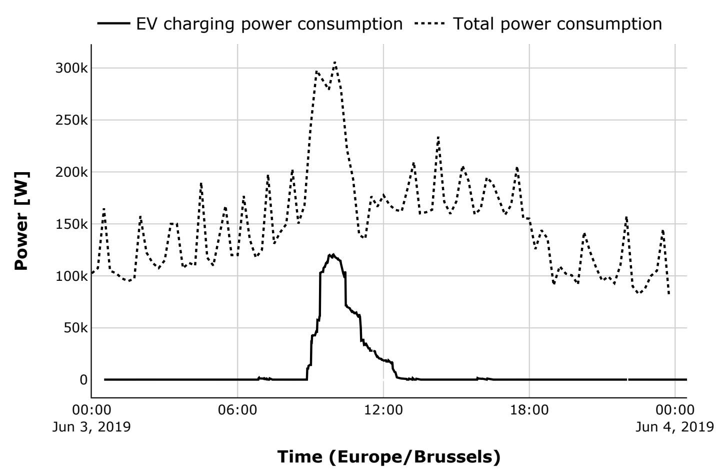


Figure 1: The typical power demand caused by EV charging in the morning can have a significant impact on the total power consumption of an office building. For example, on the morning of June 3rd, 2019, at the EnergyVille building.

In the past years, EV charging has mainly been passive or uncontrolled i.e. charging is activated immediately and at maximum charging power when an EV is plugged into a charging station. In this paper, this type of charging is referred to as business-as-usual (BAU) charging. Such uncontrolled EV charging does not exploit the inherent flexibility[1] of EV charging; typically an EV is plugged in longer than the time needed to fully charge its battery. If EV fleet charging is controlled, the flexibility harnessed can be used for a range of objectives.

Recently, several control methods have emerged for controlling the charging of EVs in order to harness flexibility for objectives such as: avoiding grid congestion problems [3], maximising self-consumption of local electricity generation [4] and load flattening [5]. These methods range from rule-based to model-based and model-free (data-driven). Rule-based methods rely on predefined rules and conditions expressed in the form “if *condition*, do *action*” statements to determine a control policy for the control agent. These rules are typically handcrafted and to guarantee an adequate performance of a rule-based controller, considerable expert knowledge is required to correctly set the threshold values, and tune the system parameters. Rule-based control has been widely used in EV charging due to its simplicity and computational efficiency for uninterrupted EV charging [6] and prevention of grid overloading [7]. However, since the rules are tailored towards a specific system and objective, the method cannot be easily generalised.

Model-based control methods, on the other hand, require an explicit definition of the system dynamics in order to establish a control policy. Model predictive control, for instance, has been extensively applied in literature for EV charging to minimize energy costs [8] and for voltage control [9, 10]. While successful, their performance relies on the accuracy of the model, and a mismatch between the model and the real system will result in sub-optimal operation. In the EV charging context, identifying a (sufficiently) accurate model is challenged by the heterogeneity of EV models and unpredictability of EV-user behavior.

In contrast to model-based methods, model-free methods do not rely on explicit knowledge of a system model. Instead, they learn a control policy from system observations collected a-priori (batch or offline learning) or through online interactions with the system. These methods are therefore data-driven, which renders them flexible and more generalisable compared to model-based and rule-based methods. The most popular model-free tech-

[1]The available EV flexibility can be described based on the number of hours that the EV charging can be delayed while meeting the user’s departure deadline and respecting the constraints on battery capacity, maximum charging rate, and additional constraints of the charging infrastructure.

nique in EV charging literature is reinforcement learning (RL) [11, 12]. Fitted Q-iteration, a batch RL technique, has been used to control EV charging for load flattening purposes [5], electricity cost savings based on day-ahead market prices [13] and long-term cost optimization [14]. Wang *et al.* [15] used an online RL algorithm, SARSA, to schedule EV charging for minimizing electricity cost. Deep Q-learning, another RL algorithm, has been used to minimize electricity cost based on real-time electricity pricing [16], minimize long-term operating cost [16], as well as for load flattening purposes [17]. The above mentioned RL algorithms are based on the standard Q-learning algorithm [18], which relies on Q-values (of state-action pairs) for the evaluation and selection of control actions, and consequently, learning of a control policy. To efficiently compute these Q-values, the action space for the learning agent is required to be finite and discrete to avoid a heavy computational burden and the curse of dimensionality. Even though the above mentioned methods employ regression algorithms such as neural networks to approximate the Q-values through a Q-function, fine-grained discrete actions lead to large (continuous) action spaces, which in turn lead to an intractable computation of Q-values. In the context of EV charging, discretizing the action space limits full exploitation of EV flexibility since the charging powers are continuous values. To allow the learning of control policies in systems with large or continuous action spaces, policy gradient methods were introduced [11].

Policy gradient techniques directly optimize a control policy without a need for Q-values to select control actions. Several policy gradient methods have been employed in EV charging literature. Yu *et al.* [19] used deep deterministic policy gradient to minimize electricity costs in a smart home with electricity generation from RES and multiple loads including EVs and HVAC systems. In [20], the authors proposed prioritised deep deterministic policy gradient for the coordination of EV fleet charging by an aggregator with the aim of maximizing profit (through vehicle-to-grid capabilities) and minimizing EV charging electricity costs. The authors showed that prioritised deep deterministic policy gradient outperforms standard Q-learning and deep Q-learning. Trust region policy optimization was used for home energy management in which the charging of an EV was controlled to minimize electricity cost [21]. Moonens and Nowé [22] used policy proximal optimization (PPO) to coordinate charging of an EV fleet for load balancing purposes. The authors showed that this method outperformed the BAU scheme by reducing the number of electricity consumption peaks caused by EV charging.

Motivated by the success of RL, and particularly policy gradient techniques for EV charging, this work builds on existing literature and proposes a novel control framework for EV charging in a work environment with the objective of maximising self-consumption of local electricity generation and minimizing electricity cost. The proposed control framework combines PPO and rule-based control allowing a quick response of the control agent to the stochastic PV generation. The main contributions of this work are summarised as follows:

- A novel control framework is proposed combining the strengths of PPO (model-free, data-driven, ability to deal with continuous actions) with those of rule-based control (low computational complexity). In the proposed framework, a RL agent learns a control policy in a low time resolution (60 minutes, 15 minutes or 5 minutes), which is refined by a rule-based controller during real-time operation (one minute time step) to ensure a more optimal real-time control. This contrasts with existing literature in which control actions are predominantly taken at an hourly resolution [5, 21].
- A demonstration of the scalability of the proposed framework by using the three-step approach introduced by Vandael *et al* [23] in which a RL agent learns the optimal aggregate charging power for an entire EV fleet. In contrast to the original work, this paper uses PPO to learn the aggregate charging power.
- Case study demonstrating that the proposed control method achieves an increase in self-consumption of local electricity generation and reductions in the net electricity costs compared to the BAU scheme. Compared to a "perfect information optimum" (PIO) strategy, the proposed control method is slightly less performant. The PIO strategy is based on sequential quadratic programming [24] and assumes full knowledge of the EV user behaviour and PV electricity generation.

The performance of the proposed framework is evaluated on a real-world data set from EnergyVille[2], a research institute in Belgium. The EV charging infrastructure at EnergyVille contains charging stations from 7 different

[2] `www.energyville.be`

brands, totalling in 27 connection points. While most charging stations are conventional 22kW AC charging stations, the setup includes an AC/DC fast-charger and a vehicle-to-grid (V2G) charging station. In total the charging stations amount to an installed EV charging power of 530kVA, while the capacity at the electrical cabinet is only 436kVA. At the moment, 12 charging stations are fully monitored and controllable via OCPP version 1.6[3], and more are expected in the short term. A custom IT infrastructure has been deployed to monitor and control the charging sessions at EnergyVille. It provides the following data: the arrival time detected when an EV is plugged in to a charging station, the maximum charging power measured three minutes into the charging session, and estimates of the departure time and total energy needed to fully charge the EV are collected as user input via a web app. A data set of all OCPP-controlled charging sessions since August 2018 is available. In addition, a 368kWp PV system is installed on the rooftop of the EnergyVille building. The energy yield of this installation can either be used for consumption within the building or injected into the grid. For the latter, a fixed fee is paid per kWh; on the other hand, consumption from the grid has a dynamic component with hourly periodicity based on the day-ahead market price. A data set is available for the PV production at EnergyVille (since April 2016). Finally, the work described in this paper also relies on the availability of day-ahead market prices and a regional PV production forecast prior to a charging session.

[3]Open Charge Point Protocol, `www.openchargealliance.org/protocols/ocpp-16/`

2. Problem description and Markov decision process

This work considers the problem of coordinating the charging of a fleet of EVs in an office building with a rooftop PV installation. The following problem description aims to be generic for this type of buildings, but employs some specific details of the EnergyVille building where needed. Charging transactions are characterised by: the arrival time T^{arr}, the departure time T^{dep}, the energy E^{req}[kWh] required to fully charge the EV, and the maximum charging power P^{lim}[kW] of the EV. The objective is to charge a fleet of N_{ev} EVs while maximising self-consumption of the locally generated electricity - from the PV installation - and minimising electricity cost; the EVs can be charged using locally generated electricity or directly from the grid. To achieve this objective, the charging power of each connected EV has to be decided at every time step based on the PV electricity generation, the electricity price, and estimates on the departure time of the EV and the energy required to fully charge the EV by that time. This decision making problem encountered at every time step can be expressed as a Markov decision process (MDP), which is the basis for formulating RL problems. However, the decision making problem is challenged by the uncertainty in the PV generation, and the arrival and departure times of the EVs.

2.1. Markov decision process formulation

A Markov decision process is characterized by: (i) a state space $\mathcal{S}$ describing the finite set of states that the system can be in, (ii) an action space $\mathcal{A}$ consisting of a finite set of possible actions that can change the state of the system, (iii) transition function f representing the system dynamics or probabilities for a stochastic state evolution, and (iv) a reward function, ρ, evaluating each state transition. Three MDP formulations - MDP_{base}, MDP_{hid} and MDP_{agg} - are presented in the following subsections. The base MDP, MDP_{base}, formulates the EV fleet charging problem described in the previous paragraph. The hidden MDP, MDP_{hid}, is similar to MDP_{base} but does not include information on the estimates of the departure time and energy required to fully charge the EV. The aggregate MDP, MDP_{agg}, builds on MDP_{base} to improve scalability to larger fleet sizes using the three-step approach introduced by Vandael *et al.* [23].

2.2. Base Markov decision process

The base Markov decision process, MDP_{base}, aims at providing optimal charging schedules for individual EVs and assumes the widest range of information to be available.

State space

The state space has three components:

- *PV component*: consists of the current PV generation $\boldsymbol{P}^{pv}$, and a forecast of the PV generation $\boldsymbol{P}^{fpv}$. $\boldsymbol{P}_t^{pv}$ contains: (i) $\boldsymbol{P}_t^{pv,past}$ a vector with the average[4] over each hour of the measured PV generation of the previous N_{past} hours, as shown in (1); and (ii) P_t^{pv}, the PV measurement at time stamp t.

$$\begin{aligned} \boldsymbol{P}^{pv} &= (\boldsymbol{P}_t^{pv,past},\ P_t^{pv}), \\ \boldsymbol{P}_t^{pv,past} &= \left[\texttt{av}^5\left(P^{pv},\ t - j \times \frac{60}{L_c},\ t - (j-1) \times \frac{60}{L_c}\right), j = N_{past}, \ldots, 1\right], \end{aligned} \tag{1}$$

  with L_c representing the length of a control time step in minutes.

  As shown in (2), the vector $\boldsymbol{P}_t^{fpv}$ contains a forecast on the PV generation in terms of $\boldsymbol{P}_t^{fpv,fut}$ (the average over each hour[6] of the forecasted PV generation P^{fpv} for the next N_{fut} hours), and $P_t^{fpv,rest}$ the average of P^{fpv} for the rest of the day, and information on the forecast of the PV generation for the past N_{past} hours, $\boldsymbol{P}_t^{fpv,past}$. In this work, N_{fut} and N_{past} are set to a value of 2 through manual tuning and experimentation by considering the trade-off between training time of the control agent and gains in electricity cost and self-consumption. Setting N_{fut} to larger values increases the uncertainty on the variables and also increases the state space dimension leading to curse of dimensionality issues and an increase in the training time.

$$\boldsymbol{P}_t^{fpv} = (\boldsymbol{P}_t^{fpv,past}, \boldsymbol{P}_t^{fpv,fut}, P_t^{fpv,rest}), \tag{2}$$

  with $\boldsymbol{P}_t^{fpv,past} = \left[\texttt{av}\left(P^{fpv},\ t - j\frac{60}{L_c},\ t-(j-1)\frac{60}{L_c}\right), j = N_{past}, ..., 1\right]$,

  $\boldsymbol{P}_t^{fpv,fut} = \left[\texttt{av}\left(P^{fpv},\ t+1+(j-1)\frac{60}{L_c},\ t+1+j\frac{60}{L_c}\right), j = 1, ..., N_{fut}\right]$,

---

[4] The average is used instead of the actual values for dimensionality reduction purposes.

[5] `av(x,t1,t2)` returns the mean value of `x(t)` between `t=t1` and `t=t2` and returns `x(t2)` if `t1`$\geq$`t2`

[6] The PV forecast values have a periodicity of 15 minutes.

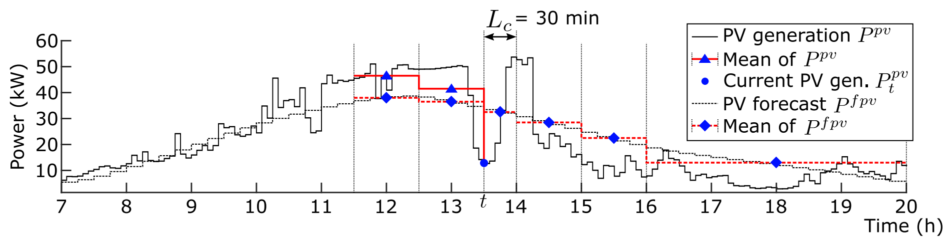


Figure 2: Example of state parameter vectors $\boldsymbol{P}_t^{pv}$ (triangles + circle) and $\boldsymbol{P}_t^{fpv}$ (diamonds) with $N_{past} = N_{fut} = 2$ and $L_c = 30$ minutes

and $P_t^{fpv,rest} = \texttt{av}\left(P^{fpv},\ t+1+N_{fut}\frac{60}{L_c},\ T\right)$, with T the final time step in the optimization horizon.

An example of $\boldsymbol{P}^{pv}$ and $\boldsymbol{P}^{fpv}$ is shown in Fig. 2.

- *Price component* $\boldsymbol{\lambda}^{cons}$*:* represents the price of importing a kilowatt-hour of energy from the grid at time t. $\boldsymbol{\lambda}^{cons}$ as shown in (3) contains the price at time step t and information on the forecasted price ($\boldsymbol{\lambda}_t^{cons,fut}$ and $\lambda_t^{cons,rest}$, defined in a similar manner as $\boldsymbol{P}_t^{fpv,fut}$ and $P_t^{fpv,rest}$).

$$\boldsymbol{\lambda}_t^{cons} = (\lambda_t^{cons},\ \boldsymbol{\lambda}_t^{cons,fut}, \lambda_t^{cons,rest}), \tag{3}$$

$\boldsymbol{\lambda}_t^{cons,fut} = \left[\texttt{av}\left(\lambda^{cons},\ t+1+(j-1)\frac{60}{L_c}, t+1+j\frac{60}{L_c}\right), j = 1, \ldots, N_{fut}\right]$ and $\lambda_t^{cons,rest} = \texttt{av}\left(\lambda^{cons},\ t+1+N_{fut}\times\frac{60}{L_c},\ T\right)$.

It is important to note that in the event where a price, λ^{inj}, is set for injecting power to the grid, this price would become part of the price component of the state space. However, in this work a constant price for injecting power to the grid is considered.

- *EV transaction component* ${}^{i}\boldsymbol{X}_t$ contains three parameters directly related to an EV charging transaction as shown in (4): (i) ${}^{i}E_t^{req}$ the energy required left to fully charge EV i (EV_i) at time t before its departure; (ii) ${}^{i}P^{lim}$ the maximum charging power for EV_i and (iii) $\Delta{}^{i}T_t^{dep} = {}^{i}T_t^{dep} - t$ the time left until departure (in number of control time steps) for EV_i at time t.

$$
{}^{i}\boldsymbol{X}_t = \begin{cases} (0,0,0), & \text{if station } i \text{ is unused at time } t, \\ \left({}^{i}E_t^{req}, {}^{i}P^{lim}, \Delta {}^{i}T_t^{dep}\right), & \text{otherwise.} \end{cases} \tag{4}
$$

In summary, at any time t, the state of the system is defined as follows:

$$
\forall s_t \in \mathcal{S}, s_t = \left({}^{1}\boldsymbol{X}_t, ..., {}^{N_{ev}}\boldsymbol{X}_t, t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}\right) \tag{5}
$$

Action space

At any time step, t, an action $a_t \in \mathcal{A}$ is a vector containing the charging powers ${}^{i}P_t$ [kW] for all the connected EVs as shown below:

$$
\forall a_t \in \mathcal{A}, a_t = \left({}^{1}P_t, ..., {}^{N_{ev}}P_t\right). \tag{6}
$$

Reward function

The reward function consists of two components C^{cons} and C^{inj} representing the cost incurred for charging the EVs from the grid and for injecting power to the grid respectively. This reward function is based on the electricity price λ^{cons} and a grid injection price λ^{inj} at time t. The total electricity cost during minute m in control time step t is given as follows:

$$
C_{t,m}^{elec} = \begin{cases} C_{t,m}^{cons} = \frac{\sum_{i=1}^{N_{EV}} {}^{i}P_{t,m} - P_{t,m}^{pv}}{60} \times \lambda_{t,m}^{cons}, & \text{if } \sum_{i=1}^{N_{EV}} {}^{i}P_{t,m} - P_{t,m}^{pv} \geq 0, \\ C_{t,m}^{inj} = \frac{\sum_{i=1}^{N_{EV}} {}^{i}P_{t,m} - P_{t,m}^{pv}}{60} \times \lambda^{inj}, & \text{otherwise,} \end{cases} \tag{7}
$$

where ${}^{i}P_{t,m}$ is the charging power for EV_i and $P_{t,m}^{pv}$ is the PV generation at minute m of time step t.

Based on the cost incurred per minute, the negative reward function for each state transition is defined as shown in (8).

$$
\rho(s_t, a_t, s_{t+1}) = -\sum_{m=0}^{L_c - 1} C_{t,m}^{elec} \tag{8}
$$

In this work, we focus on a data-driven RL algorithm, as such, the definition of a transition function for the system dynamics is not required. However, the state of each connected EV is updated as follows:

$$ {}^{i}E_t^{ch} = \sum_{m={}^{i}M_t^{start}}^{{}^{i}M_t^{end}} {}^{i}P_{t,m}/60 \tag{9} $$

$$ {}^{i}\boldsymbol{X}_{t+1} = \begin{cases} (0,0,0), & \text{if } {}^{i}E_t^{req} - {}^{i}E_t^{ch} \leq 0 \text{ or } \Delta {}^{i}T_t^{dep} - 1 \leq 0, \\ \left({}^{i}E_t^{req} - {}^{i}E_t^{ch}, {}^{i}P^{lim}, \Delta {}^{i}T_t^{dep} - 1\right), & \text{otherwise,} \end{cases} \tag{10} $$

where ${}^{i}E_t^{ch}$ is the energy charged by EV$_i$ during time step t. New EVs that arrived between t and $t+1$ are added to ${}^{i}\boldsymbol{X}_{t+1}$. The remaining state parameters $\boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}$ are updated by reading the data from the database.

2.3. MDP with unknown SoC and departure time

As mentioned earlier, the availability of estimates of T^{dep} and E^{req} is based on the willingness of the EV users to provide these inputs in practice. They can be termed as hidden parameters as they cannot be measured directly without user interaction, and as such, are hidden from the learning agent. Therefore, to allow learning adequate control policies in scenarios without user interaction, MDP$_{\text{hid}}$ is proposed in this section. The definition of MDP$_{\text{hid}}$ is similar to that of MDP$_{\text{base}}$. However, the state space is different since ${}^{i}E_t^{req}$ and ${}^{i}T_t^{dep}$ are not included.

State space

Compared to MDP$_{\text{base}}$, the hidden state parameters ${}^{i}E_t^{req}$ and ${}^{i}T_t^{dep}$ in the state s_t are replaced by two known state parameters: (i) ${}^{i}E_t^{ch,ses}$ the total energy already charged in the current session of EV_i at time t, and (ii) the arrival time ${}^{i}T^{arr}$. The state at time t, s_t, is now defined as:

$$ s_t = ({}^{1}\boldsymbol{X}_t, ..., {}^{N_{ev}}\boldsymbol{X}_t, t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}) \tag{11} $$

$$ \text{with } {}^{i}\boldsymbol{X}_t = \begin{cases} (0,0,0), & \text{if station } i \text{ empty at time } t \\ ({}^{i}E_t^{ch,ses}, {}^{i}P^{lim}, {}^{i}T^{arr}), & \text{otherwise.} \end{cases} \tag{12} $$

The remaining state parameters ${}^{i}P^{lim}, t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}$ are the same as in MDP$_{\text{base}}$.

Action space

The action space is the same as MDP_{base}.

Reward function

The reward function consists of two terms as shown in (13).

$$\rho(s_t, a_t, s_{t+1}) = -\sum_{m=0}^{L_c-1} C_{t,m}^{elec} - \sum_{i=1}^{N_{ev}} {}^{i}C_t^{anx} \tag{13}$$

$$\text{with } {}^{i}C_t^{anx} = \begin{cases} 0, & \text{if } \Delta {}^{i}T_t^{dep} > 0, \\ K_1 \times \frac{({}^{i}E_t^{req})^{K_2}}{({}^{i}E^{req})^{K_3}} \times \bar{\lambda}^{cons}, & \text{if } \Delta {}^{i}T_t^{dep} = 0, \end{cases} \tag{14}$$

where $\sum_{m=0}^{L_c-1} C_{t,m}^{elec}$ is the electricity cost in (7) and ${}^{i}C_t^{anx}$ is the "range anxiety" cost, a penalty for not charging EV_i with its ${}^{i}E^{req}$ [25]. $\frac{{}^{i}E_t^{req}}{{}^{i}E^{req}}$ is the fraction of uncharged energy at time t. The coefficients K_1, K_2 and K_3 are hyperparameters and $\bar{\lambda}^{cons}$ is the average of the (dynamic) electricity consumption price profile of the current day. In this case, the objective of the RL algorithm is to minimize the charging cost and the uncharged energy fraction in each session.

The coefficient K_1 weighs the trade-off between the charging cost and the average amount of uncharged energy. The coefficients K_2 and K_3 are exponents that allow testing the performance with different types of the anxiety function. For example, the anxiety cost with $K_2 = 2$ and $K_3 = 2$ is proportional to the square of the fraction of the uncharged energy, while with $K_2 = 1$ and $K_3 = 1$ the relationship is linear. Notice that, to compute the anxiety at time t, knowledge of ${}^{i}E_t^{req}$ and ${}^{i}T_t^{dep}$ is required. However, that knowledge is only required in the training phase of the RL algorithm when ${}^{i}E_t^{req}$ and ${}^{i}T_t^{dep}$ are readily available from historical data. During policy execution, computation of the reward is not required - since we will focus on a policy gradient RL algorithm. Thus, knowledge of ${}^{i}E_t^{req}$ and ${}^{i}T_t^{dep}$ is not required during policy execution.

For each connected EV, its state is updated as follows:

$${}^{i}\boldsymbol{X}_{t+1} = \begin{cases} (0,0,0), & \text{if battery full or } \text{EV}_i \text{ has departed}, \\ \left({}^{i}E_t^{ch,ses} + {}^{i}E_t^{ch}, {}^{i}P^{lim}, \Delta {}^{i}T^{arr}\right), & \text{otherwise}, \end{cases} \tag{15}$$

where ${}^{i}E_t^{ch}$ is the energy charged by EV_i during slot t as defined in (9). Similar to the base MDP, new EVs that arrive between t and $t+1$ are added to ${}^{i}\boldsymbol{X}_{t+1}$, and the remaining state parameters $t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}$ are also updated.

2.4. MDP with aggregated state-action space

To improve the scalability of the base MDP for larger fleet sizes, $\mathrm{MDP}_{\mathrm{agg}}$ is proposed. This MDP builds on the three-step method for smart EV charging proposed in [23]. This approach consists of three steps: (i) an aggregation step in which the individual EV charging constraints are established in the form of priorities and aggregated; (ii) an optimization step that uses the aggregated constraints to compute a collective charging plan for all the EVs, with the aim of maximizing self-consumption and minimising electricity costs; and (iii) a real-time control step dividing and dispatching the charging plan to all the EVs. In this aggregate MDP, the reward function is the same as that in Section 2.2, therefore only the state and action spaces are described below.

State space

The state s_t at time t is defined as:

$$s_t = (\boldsymbol{Z}_t, t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}), \tag{16}$$

where $\boldsymbol{Z}_t$, as shown in (17), is the aggregated fleet state, which is obtained through manual feature extraction consisting of the total fleet required energy and the total fleet maximum charging power with ${}^{i}P^{lim} = 0$ if station i is unused. The state parameters $t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}$ are the same as those in the base MDP.

$$\boldsymbol{Z}_t = \left(\sum_{i=1}^{N_{ev}} {}^{i}E_t^{req}, \sum_{i=1}^{N_{ev}} {}^{i}P^{lim} \right) \tag{17}$$

It is important to note that even though we do not consider the dynamics of the system, the aggregate fleet state can be updated as follows:

$$\boldsymbol{Z}_{t+1} = \left(\sum_{i=1}^{N_{ev}} {}^{i}E_t^{req} - {}^{i}E_t^{ch},\ \sum_{i=1}^{N_{ev}} {}^{i}P^{lim} \right), \tag{18}$$

where ${}^{i}E_t^{ch}$ is the energy charged by EV_i during slot t as defined in (9). New EVs that arrive between t and $t+1$ are added to $\boldsymbol{Z}_{t+1}$, and the remaining state parameters $t, \boldsymbol{P}_t^{pv}, \boldsymbol{P}_t^{fpv}, \boldsymbol{\lambda}_t^{cons}$ are also updated by reading the values from the database.

Action space

An action $a_t \in \mathcal{A}$ at time step t represents an aggregate charging power P^a for the entire fleet as shown in (19).

$$\forall a_t \in \mathcal{A}, a_t = P_t^a = \sum_{i=1}^{N_{ev}} {}^iP_t \tag{19}$$

This aggregate charging power is divided and dispatched to all the EVs using a heuristic dispatch described below.

Heuristic dispatch

The aggregate charging power is divided to all the EVs in the fleet using a heuristic dispatch based on (20). This dispatch also ensures that each EV leaves with its required energy charged and is not charged with a power greater than its maximum charging power.

$$\boldsymbol{P}_t^b = \left({}^1P_t^b, ..., {}^{N_{ev}}P_t^b\right) \tag{20}$$

$$\text{s.t} \quad {}^iP_t^{b,min} \leq {}^iP_t^b \leq {}^iP_t^{b,max}, \quad \forall i \in [1, N_{ev}] \tag{21}$$

$${}^iP_t^{b,min} = \max\left(0,\ ({}^iE_t^{req} - (\Delta {}^iT_t^{dep} - 1) \times {}^iP^{lim}) \times \frac{60}{{}^iM_t^{end} - {}^iM_t^{start}}\right), \tag{22}$$

$$\text{with } {}^iP_t^{b,max} = \min\left({}^iP^{lim},\ {}^iE_t^{req} \times \frac{60}{{}^iM_t^{end} - {}^iM_t^{start}}\right) \tag{23}$$

${}^iP_t^{b,min}$ is the minimum charging power required to guarantee that EV_i leaves with its battery fully charged, i.e. to guarantee ${}^iE_t^{req} = 0$ when $\Delta {}^iT_t^{dep} = 0$. Overcharging - charging above maximum charging power - is prevented by the maximum charging power ${}^iP_t^{b,max}$, which is limited by ${}^iP^{lim}$ and ${}^iE_t^{req}$. ${}^iM_t^{start}$ and ${}^iM_t^{end}$ are the first minute and last minute respectively of EV_i in time step t where $0 \leq {}^iM_t^{start} < L_c$ and $0 < {}^iM_t^{end} \leq L_c$ and are computed as follows:

$$ {}^{i}M_t^{start} = \begin{cases} \lfloor ({}^{i}T^{arr} - t) \times L_c \rfloor, & \text{if } t \leq {}^{i}T^{arr} < t+1, \\ 0, & \text{otherwise.} \end{cases} \quad (24) $$

$$ {}^{i}M_t^{end} = \begin{cases} \lceil ({}^{i}T^{dep} - t) \times L_c \rceil, & \text{if } t \leq {}^{i}T^{dep} < t+1, \\ L_c, & \text{otherwise.} \end{cases} \quad (25) $$

The operation of this heuristic dispatch can be described as "least-flexible first" scheduling. EVs are assigned partitions of the aggregate charging power P^a in order of priority. EVs with a lower flexibility are given a higher priority. The flexibility ${}^{i}\tau$ of each EV_i is calculated according to (26).

$$ {}^{i}\tau := \Delta {}^{i}T^{dep} - 1 - \frac{{}^{i}E^{req} - {}^{i}E^{ch}}{{}^{i}P^{lim}} \times \frac{60}{L_c}. \quad (26) $$

This priority represents the number of time steps until the next time step $t+1+{}^{i}\tau$ at which ${}^{i}P^{b,min}_{t+1+{}^{i}\tau} > 0$ (to ensure EV_i leaves with ${}^{i}E_t^{req} = 0$), assuming EV_i charges ${}^{i}E^{ch}$ at time step t. To illustrate this priority computation for $L_c = 60$ minutes, consider an EV with $\Delta {}^{i}T^{dep} = 8$ time steps, ${}^{i}E^{req} = 10$ kWh, ${}^{i}E^{ch} = 0$ kWh, ${}^{i}P^{lim} = 5$ kW that needs exactly 2 time steps to fully charge at a power of ${}^{i}P^{lim}$. The earliest time step when its ${}^{i}P^{b,min}_{t+1+{}^{i}\tau} > 0$ is thus ${}^{i}\tau = 8-1-2 = 5$ slots from $t+1$. The dispatch algorithm assigns partitions of P_t^a heuristically with the aim to maximize the minimum value of ${}^{i}\tau$:

$$ \text{maximize } \min_i {}^{i}\tau. \quad (27) $$

Reward function

The reward function is the same as that of MDP_{base}.

It is important to note that even though the MDPs described above do not explicitly take into account the system constraints, which come in the form of ensuring that the EV is fully charged before its departure and that the EV is not charged at a power greater than its maximum charging power, this work proposes using a backup controller to ensure these constraints are respected. This backup controller is described in the next section.

3. Algorithms

To solve the EV fleet charging problem described in Section 2, a policy gradient RL algorithm, proximal policy optimization (PPO) [26], is used. The goal of the learning agent is to find a (parameterised) control policy π (i.e. a mapping from a given or perceived state to the action that has to be taken in that state, $\pi : \mathcal{S} \rightarrow \mathcal{A}$), which maximises the return over the optimization horizon from some initial state s_0 as shown in (28).

$$G^{\pi}(s_0) = \sum_{t=1}^{T-1} \gamma^t \rho(s, \pi(s)), \tag{28}$$

where $\gamma \in [0; 1]$ is a discount factor that takes into account the uncertainty in the future reward and T is the length of the finite optimisation horizon. This return is the discounted cumulative reward along a trajectory generated by the policy.

3.1. Proximal policy optimization

Proximal policy optimization is a policy gradient RL algorithm based on the actor-critic algorithm [11] that directly optimises a parameterised and differentiable policy. The policy must be differentiable with respect to its parameters to allow computation of the gradient required for the policy parameter updates. Typically, the policy is represented by a neural network and is expressed as follows:

$$\pi(a|s, \theta) = Pr(a|s, \theta), \tag{29}$$

where θ represents the weights of the neural network. The goal is therefore to find the values of θ that maximise the return G.

The algorithm uses the clipped surrogate objective - the probability ratio pr of the old policy and new policy - with the aim of providing a more stable update of the policy parameters [26]. The unconstrained objective function that PPO aims to maximize is as shown in (30).

$$L^{ORIG}(\theta) = \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{\Lambda}_t \right] = \mathbb{E} \left[pr_t(\theta) \hat{A}_t \right], \tag{30}$$

where $pr_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ and $\hat{\Lambda}_t$ is an estimate of the advantage ($\Lambda(s_t, a_t) = Q(s, a) - V(s)$, $Q(s, a)$ is the state-action value function and $V(s)$ is the state value function). $L^{ORIG}(\theta)$ can still lead to large gradient updates and consequently, instability during learning. This is remedied by using the clipped surrogate objective shown in (31).

$$L_t^{CLIP}(\theta) = \mathbb{E}_t \left[\min \left(\rho_t(\theta)\hat{\Lambda}_t, \text{clip}\left(\rho_t(\theta), 1-\delta, 1+\delta\right)\hat{\Lambda}_t \right) \right], \tag{31}$$

where $\delta \in [0.1, 0.3]$ according to [26].

The clipped surrogate objective is further augmented for applying it to a neural network architecture with shared parameters for representing the policy and value functions[7]. Typically, the policy and value network share the first few hidden layers, which perform feature extraction of the state space. Additionally, an entropy term is included to the objective to increase exploration and as such more coverage of the state space. The new objective is as shown in (32).

$$L_t^{CLIP2}(\theta) = \mathbb{E}_t \left[L_t^{CLIP}(\theta) - c_1 \left(V_\theta(s_t) - V_t^{\text{targ}} \right)^2 + c_2 H[\pi_\theta](s_t) \right], \tag{32}$$

where c_1 and c_2 are hyper-parameters and H is an entropy measure. The resulting PPO algorithm is described in Algorithm 1. The algorithm parallelises the sampling of the agent-environment interactions - by using N parallel actors - and uses multiple epochs of stochastic gradient ascent per policy update. Parallel sampling significantly speeds up training times by using parallel processors while the ability to use multiple epochs when updating the neural network increases sample efficiency.

3.2. Backup controller

The backup controller is an overrule mechanism that ensures that the system constraints are respected. Recall that in the context of this paper, these constraints are the charging power limits of the EV, and the need to fully charge the EV before its departure. The backup controller therefore clips ${}^{i}P_t$ - the charging power or action suggested by the RL agent - at each time step t for each EV_i according to (33).

[7] The value function is represented by a neural network for function approximation to make it more generalisable over unseen state(-action pairs).

Algorithm 1: Proximal policy optimization [26]

```
Input : policy parameters θ_0, clipping threshold δ, initial
        network parameters θ_0
for k := 0, 1, ... do
    for actor := 1, ..., N do
        Run policy π_k in environment for T time steps;
        Compute rewards r̂_t;
        Compute advantage estimates Λ̂_t;
    Compute policy update θ_{k+1} := arg max_θ L^{CLIP2}(θ)
    with K epochs of mini-batch stochastic gradient ascent, where
    L^{CLIP2}(θ) := E_{τ∼π_k}[Σ_{t=0}^{T} L_t^{CLIP2}(θ)];
```

$$
{}^{i}P^{b}_{t} = \begin{cases} {}^{i}P^{b,min}_{t}, & \text{if } {}^{i}P_{t} \leq {}^{i}P^{b,min}_{t}, \\ {}^{i}P_{t}, & \text{if } {}^{i}P^{b,min}_{t} < {}^{i}P_{t} \leq {}^{i}P^{b,max}_{t}, \\ {}^{i}P^{b,max}_{t}, & \text{if } {}^{i}P_{t} > {}^{i}P^{b,max}_{t}. \end{cases} \tag{33}
$$

Recall that ${}^{i}P^{b,min}_{t}$ is the minimum charging power required to guarantee that EV_i leaves with the required energy charged, i.e. ${}^{i}E^{req}_{t} = 0$ when $\Delta{}^{i}T^{dep}_{t} = 0$, as shown in (22), and ${}^{i}P^{b,max}_{t}$ is limited by ${}^{i}P^{lim}$ and ${}^{i}E^{req}_{t}$, as shown in (23).

It is important to note that when EV_i, arrives in control slot t ($t \leq {}^{i}T^{arr} < t+1$), it is entirely controlled by the backup controller starting from ${}^{i}T^{arr}$ up to $t+1$. No action is taken by the agent (${}^{i}P = 0$), and ${}^{i}M^{start}_{t}$ is set to $\lfloor({}^{i}T^{arr} - t) \times L_c\rfloor$ (24). The backup controller determines ${}^{i}P^{b}$ by clipping the ${}^{i}P = 0$ using (33) to ensure EV_i can still charge its required energy during the remainder of the session starting from $t+1$. Therefore, the backup controller ensures that the charging power for EV does not exceed ${}^{i}P^{lim}$ and that each EV_i is charged with exactly its ${}^{i}E^{req}$ at departure.

3.3. Real-time controller

The output of the backup controller, ${}^{i}P^{b}_{t}$ for $i = 1...N_{ev}$, is a charging power for each EV for control time step t. On cloudy days the PV electricity generation can fluctuate rapidly during the entire time step. As a result, even when the total charging power in a control time step, $\sum_{i=1}^{N_{ev}} {}^{i}P^{b}_{t}$, is equal to the mean PV generation in the same control time step, $\texttt{av(}P^{pv}, t, t+1\texttt{)}$, the total charging cost for that slot C_t^{elec} is not zero because the PV generation fluctuations incur extra electricity consumption and injection costs. Hence, an algorithm that can learn on a fine timescale and can adapt the charging power to the rapidly fluctuating PV generation has a lower bound on the minimum charging cost it can achieve than an algorithm that learns on a lower time resolution. One solution is to learn on a fine timescale by reducing L_c, which results in charging decisions being taken more frequently. However, the time to simulate each episode is inversely proportional to L_c. A decrease in L_c thus leads to an increase in training time.

Instead of decreasing L_c, we propose a hybrid solution in which the control period L_c is set relatively large (e.g. 60 minutes) and the RL algorithm is combined with a rule-based real-time controller (RTC) that dynamically adapts the charging powers from the low resolution RL algorithm to the current real-time PV generation on a higher resolution of one minute. The RTC takes the output of the backup controller, ${}^{i}P_t^b$ for $i = 1...N_{ev}$, and for each minute m of time step t determines a real-time charging power ${}^{i}P_{t,m}^r$ for each EV_i by solving the following optimization problem:

$$\begin{aligned} \min \quad & \sum_{m=0}^{L_c-1} \left| P_{t,m}^{pv} - \sum_{i=1}^{N_{ev}} {}^{i}P_{t,m}^r \right|, \\ \text{s.t.} \quad & \sum_{m=0}^{L_c-1} {}^{i}P_{t,m}^r / L_c = {}^{i}P_t^b \; \forall i \in [1, N_{ev}], \\ & {}^{i}P_{t,m}^r \leq {}^{i}P^{lim} \forall i \in [1, N_{ev}], \;\; \forall m \in [0, L_c - 1]. \end{aligned} \tag{34}$$

By solving the above optimization problem through a set of manually defined rules, the RTC computes the power schedule for each EV_i for each minute m throughout the duration of time step t according to (35).

$${}^{i}\boldsymbol{P}_t^r = \left({}^{i}P_{t,m}^r = {}^{i}P_t^b \quad \text{for } {}^{i}M_t^{start} \leq m < {}^{i}M_t^{end} \right). \tag{35}$$

This ensures that the total charging power stays as close as possible to the PV generation in each minute of the time step while ensuring that the total energy for charging the EV during that time step is equal to the energy suggested by the backup controller. The RTC also ensures that the charging power for each EV in each minute m does not exceed the absolute maximum charging power ${}^{i}P^{lim}$ for that EV.

The RTC algorithm is described in Algorithm 2. For readability, the subscript t is dropped for most variables in the algorithm. For each EV to be charged, ${}^iP^{r,min}$, ${}^iP^{r,max}$ and ${}^iE^{rem}$ are calculated. The variables ${}^iP^{r,min}$ and ${}^iP^{r,max}$ are minimum and maximum charging powers that apply to each minute m during time step t, while ${}^iE^{rem}$ is the remaining energy to be charged in the current control time step as specified by ${}^iP^b$ (the backup controller output). ${}^iP^{r,min}$ and ${}^iP^{r,max}$ are limited by the charging flexibility factor κ, a hyper-parameter that dictates by how much the charging power in each minute can deviate from ${}^iP^b$. A value of $\kappa = 1$ is equivalent to not using the RTC.

Figure 3 illustrates the interactions between the different algorithms in the proposed control scheme.

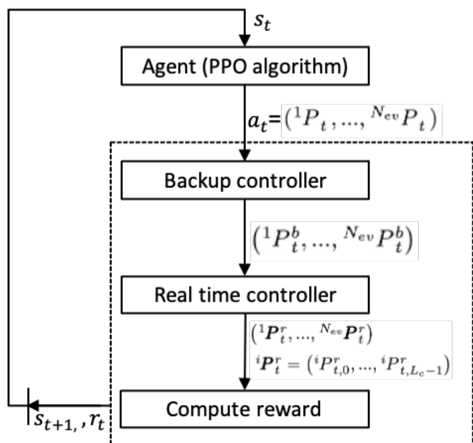


Figure 3: An illustration of the interaction between the different algorithms. The agent uses the PPO algorithm for action selection. Everything outside the agent is considered as its environment.

Algorithm 2: Real-Time Controller

Input : $\left({}^{i}P^{b},\, {}^{i}M^{start},\, {}^{i}M^{end} \text{ for } i = 1...N_{ev}\right)$, P^{pv}_{m} for $m = 0...L_c - 1$

Parameter: Charging flexibility κ

Output : ${}^{i}\boldsymbol{P}^{r}_{t} = ({}^{i}P^{r}_{t,0}, ..., {}^{i}P^{r}_{t,L_c-1})$

`EVsToCharge` $\coloneqq$ set of all i where ${}^{i}P^{b} > 0$;

for i *in* ***EVsToCharge*** **do**

  ${}^{i}P^{r,max} \coloneqq \min({}^{i}P^{lim}, {}^{i}P^{b} \times \kappa)$;

  ${}^{i}P^{r,min} \coloneqq {}^{i}P^{b}/\kappa$;

  ${}^{i}E^{rem} \coloneqq {}^{i}P^{b} \times ({}^{i}M^{end} - {}^{i}M^{start})/60$;

for $m \coloneqq 0...L_c - 1$ **do**

  `EVsThisMinute` $\coloneqq$ set of all i where ${}^{i}M^{start} \leq m < {}^{i}M^{end}$;

  for i *in* ***EVsToCharge*** $\cap$ ***EVsThisMinute*** **do**

    ${}^{i}P^{r,max}_{m} \coloneqq \min\left({}^{i}E^{rem} \times 60,\; {}^{i}P^{r,max}\right)$;

    ${}^{i}P^{r,min}_{m} \coloneqq$ $\operatorname{clip}\left({}^{i}E^{rem} \times 60 - ({}^{i}M^{end} - m - 1) \times {}^{i}P^{r,max},\; {}^{i}P^{r,min},\; {}^{i}P^{r,max}_{m}\right)$;

  $P^{min,\Sigma}_{m} \coloneqq \sum_{i=1}^{N_{ev}} {}^{i}P^{r,min}_{m}$;

  $P^{max,\Sigma}_{m} \coloneqq \sum_{i=1}^{N_{ev}} {}^{i}P^{r,max}_{m}$;

  for i *in* ***EVsToCharge*** $\cap$ ***EVsThisMinute*** **do**

    if $P^{max,\Sigma}_{m} - P^{min,\Sigma}_{m} == 0$ **then**

      ${}^{i}P^{r}_{t,m} \coloneqq P^{r,min}_{m}$;

    else

      ${}^{i}P^{r}_{t,m} \coloneqq$ $\operatorname{clip}\left({}^{i}P^{r,min}_{m} + \frac{P^{pv}_{m} - P^{min,\Sigma}_{m}}{P^{max,\Sigma}_{m} - P^{min,\Sigma}_{m}}({}^{i}P^{r,max}_{m} - {}^{i}P^{r,min}_{m}),\, {}^{i}P^{r,min}_{m},\, {}^{i}P^{r,max}_{m}\right)$;

    ${}^{i}E^{rem} \coloneqq {}^{i}E^{rem} - {}^{i}P^{r}_{t,m}/60$;

return ${}^{i}\boldsymbol{P}^{r}_{t}$;

4. Case study and simulation results

The proposed framework is evaluated using the EnergyVille office building as a case study. The various MDPs are evaluated on (i) self-consumption of local PV-generated electricity and net electricity costs, and (ii) scalability in terms of fleet size.

4.1. Case study

This work considers real-world data sets on energy consumption and generation at EnergyVille, complemented with data from Belgian electricity markets and regional PV forecasts. More specifically:

- PV generation measurements $P^{pv,tot}$ of the EnergyVille rooftop PV installation measured at 5 minute intervals
- PV forecast with a 15 minute time step for the province of Limburg, Belgium (location of EnergyVille)[8]
- EV charging transactions at EnergyVille: 586 valid historical charging sessions across 171 days collected between 08/08/2018 and 20/09/2019. For each EV transaction, T_{arr} is known when the EV is plugged in, P^{lim} is detected during the first few minutes of charging, T^{dep} and E^{req} are extracted from the historical data set.
- EV power consumption ${}^{i}P^{hist}$ of each historical charging session at EnergyVille, measured every 20 seconds
- Historical grid consumption P^{cons} and injection P^{inj}[9] of the EnergyVille building, collected every 15 minutes
- A grid injection tariff of $\lambda^{inj} = 1.46$€ /MWh as the rooftop PV installation at EnergyVille is larger than 10kVA.
- A TOU grid consumption price[10] λ^{Belpex} with a one hour periodicity is used to compute the grid consumption prices λ^{cons} as shown in (36).

[8] Solar-PV power forecasting for Belgium: `https://www.elia.be/en/grid-data/power-generation/solar-pv-power-generation-data`

[9] P^{inj} is the surplus PV generation injected to the grid.

[10] Belgian day-ahead market prices: `https://transparency.entsoe.eu/transmission-domain/r2/dayAheadPrices/show`

$$\lambda^{cons} = (\lambda^{\text{Belpex}} + 0.045 \text{ €/kWh}) \times 1.21, \tag{36}$$

where 0.045 €/kWh are the estimated grid tariffs, and a 21% VAT is charged. By including taxes in λ^{cons} an estimate of the actual cost savings for EnergyVille is obtained in the experiments.

It is worth noting that in our simulations a scaled version of the PV generation, $P^{pv} = P^{pv,scaled} \equiv 0.2 \times P^{pv,tot}$ is used to filter out the influence of the stochastic electricity consumption of the rest of the building. This scaling expresses a hypothetical scenario where a small PV installation is available solely for EV charging.

Also, 98% of the charging transactions in the data set occur between 7:00 and 20:00. Therefore, 7:00 and 20:00 are set as the start and end times of each episode respectively. The algorithms are tested using three control time steps: $L_c = 5$, $L_c = 15$ and $L_c = 60$ minutes. These control time steps have been selected by considering the 15 minutes time step for the imbalance electricity market, the 60 minutes time step of the day-ahead electricity market, and in order to approach real-time operation, a time step of 5 minutes.

To evaluate the performance of the proposed control framework the simulation results are compared with those from the business as usual (BAU) and "perfect information optimum" (PIO) strategies. Recall that the BAU strategy is equivalent to passive charging, where each EV is charged immediately when it is plugged in to the charging station at its maximum power ${}^iP^{lim}$ until the required energy ${}^iE_t^{req}$ reaches zero. The PIO strategy assumes complete knowledge for the entire day of all EV arrival and departure times, required energy, maximum power and the PV generation and electricity prices. The problem is formulated as a constrained nonlinear optimisation problem as shown in (37) and solved using the sequential least-squares quadratic programming algorithm [24, 27]. Due to limited computational resources, $L_c = 15$ minutes is used. The BAU and PIO are selected as the baselines because; (i) the BAU is the strategy that is used in most charging stations, and (ii) the PIO provides a theoretical baseline considering a scenario in which all the information on the different system variables is available. These baselines provide a best- and worst-case comparison.

$$
\begin{aligned}
&\min_{\boldsymbol{P}^r} C^{day} = \sum_{t=0}^{T-1} C_t^{elec} \\
&\text{s.t } C_t^{elec} = \begin{cases} C_t^{cons} = \sum_{i=1}^{N_{EV}} ({}^iP_t^r - P_t^{pv}) \times (L_c/60) \times \lambda_t^{cons}, & \text{if } \sum_{i=1}^{N_{EV}} {}^iP_t^r - P_t^{pv} \geq 0, \\ C_t^{inj} = \sum_{i=1}^{N_{EV}} (P_t^{pv} - {}^iP_t^r) \times (L_c/60) \times \lambda_t^{inj}, & \text{otherwise}, \end{cases} \\
&\quad 0 \leq {}^iP_t^r \leq {}^iP^{lim} \quad \forall i \in [1, N_{ev}],\ \forall t \in [0, T-1], \\
&\qquad \sum_{i=1}^{N_{ev}} \sum_{t={}^iT^{arr}}^{{}^iT^{dep}} {}^iP_t^r \times (L_c/60) = {}^iE^{req}.
\end{aligned} \tag{37}
$$

The neural network used to represent the actor-critic in the PPO algorithm consists of a first hidden layer with 128 nodes, shared between the actor and the critic. Both networks contain two hidden layers with 64 nodes each. The number of layers and nodes are obtained based on [22] and [26]. The `tanh` activation function is used. A representation of the actor-critic network is shown in Fig. 4.

4.2. Simulation results

The performance of the proposed control strategy is evaluated by considering two simulation experiments. The first experiment evaluates the performance of the control strategy for the different MDP formulations while the second experiment investigates the scalability of the control framework. The net electricity cost per day (an optimisation horizon of one day is used) is considered as the key performance indicator.

4.2.1. Experiment 1: performance evaluation of MDPs

This experiment compares the performance of the control framework for the three MDP formulations and investigates the influence of the time step L_c and the RTC on the performance. The MDPs are tested for L_c=5, L_c=15 and L_c=60 minutes, and each of these instances is tested without the RTC and with the RTC (for $\kappa = 1.5$ and with $\kappa = 5.0$). For each instance, the training/testing loop is executed for 5×10^7 agent-environment interactions. The remaining MDP hyper-parameters are set as follows: $N_{past} = N_{fut} = 2$, $K_1 = 5.0 \times 10^4$, $K_2 = 1$, $K_3 = 1$ (MDP$_{\text{hid}}$).

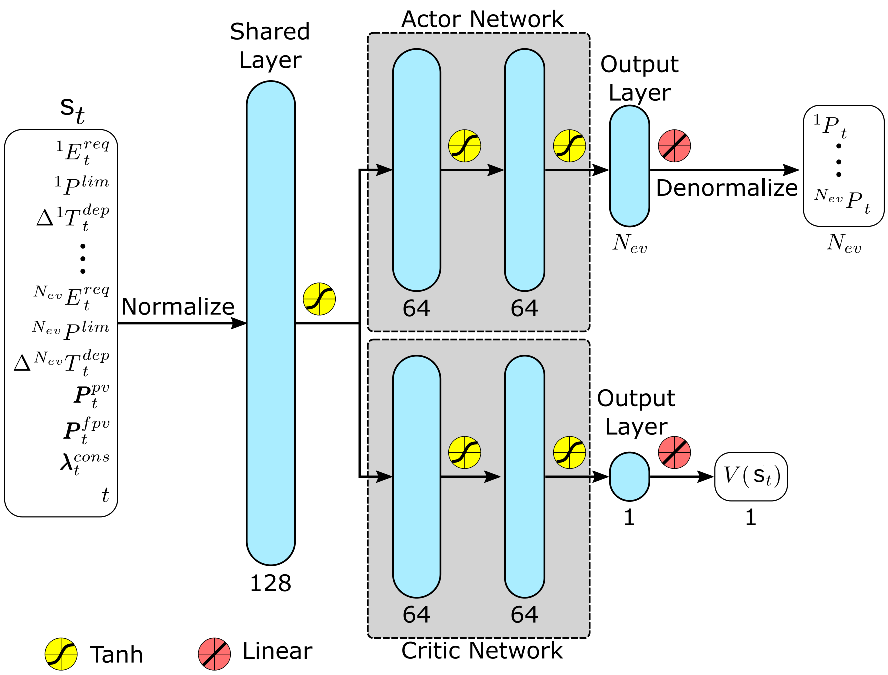
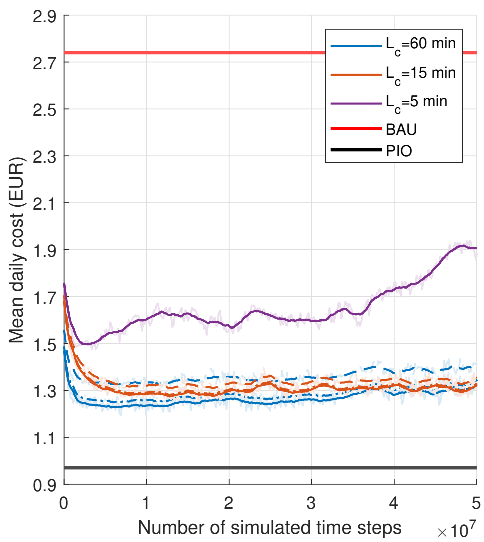
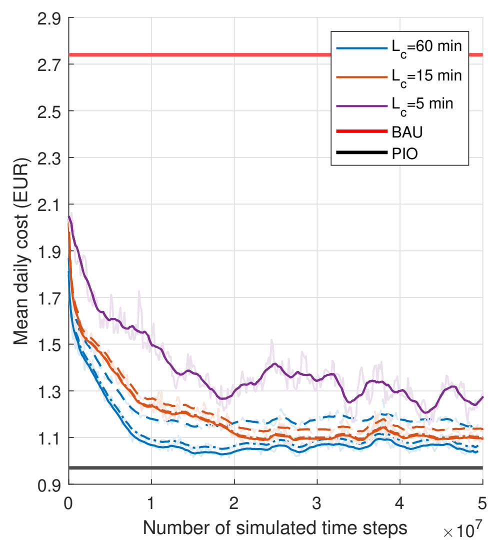


Figure 4: Illustration of the actor-critic network. This network is used for action selection - the policy by the actor - and for estimating the value function - critic.

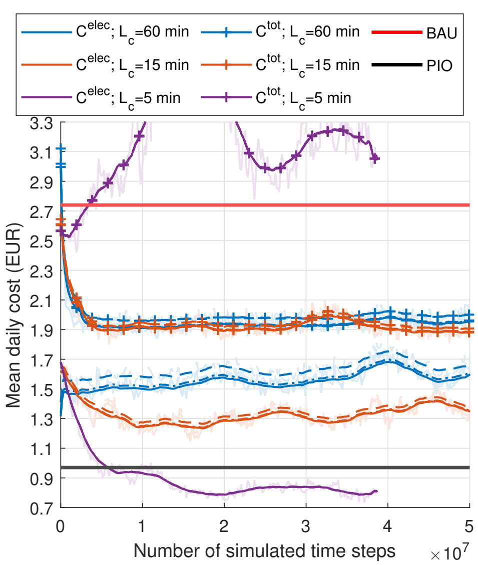
The simulation results are presented in Fig. 5. These results are expressed in terms of the total cost $C^{tot} = C^{cons} + C^{inj} + C^{anx}$ and the electricity cost $C^{elec} = C^{cons} + C^{inj}$. For MDP_{base} and MDP_{agg}, $C^{anx} = 0$. MDP_{agg} obtains the lowest electricity cost and has the most stable learning curve while MDP_{base} tends to converge quickly to a sub-optimal local minimum. The resulting electricity cost for MDP_{agg} is 0.2€ lower than that of MDP_{base} (for $L_c = 60$). For MDP_{hid}, the trade-off between electricity cost and the fraction of uncharged energy (Fig. 5c and 5d) depends heavily on L_c. When $L_c = 60$ minutes, there is a 0.2€ increase in electricity cost compared to MDP_{base}, and 2% of the EVs leave with 25% of their required energy not met. It is worth noting that the EVs leave fully charged in the case of MDP_{base} and MDP_{agg} as knowledge of the required energy and departure times of the EVs allows the agent to obtain (near) optimal schedules. Moreover, this knowledge of required energy and departure times allows the backup controller to override the actions of the learning agent and ensure that the EVs are fully charged before departure.



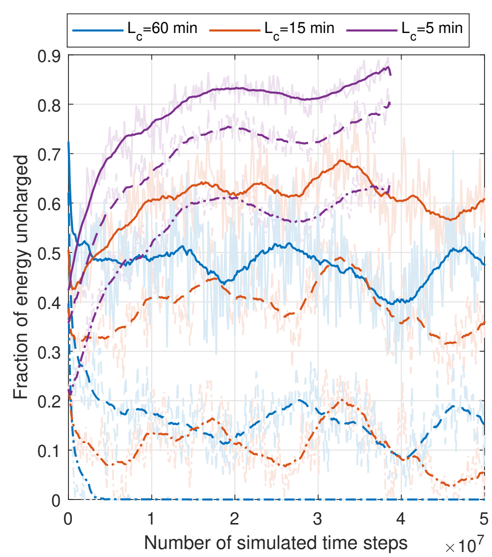
(a) Evolution of C^{elec} for MDP_{base} (solid = RTC with $\kappa = 5.0$, dash-dotted = RTC with $\kappa = 1.5$, dash = no RTC)



(b) Evolution of C^{elec} for MDP_{agg} (solid = RTC with $\kappa = 5.0$, dash-dotted = RTC with $\kappa = 1.5$, dash = no RTC)



(c) Evolution of C^{elec} and C^{tot} for MDP_{hid} (solid = RTC with $\kappa = 5.0$, dash-dotted = RTC with κ=1.5, dash = no RTC)



(d) Evolution of 99- (solid), 98- (dashed) and 96-percentile (dash-dotted) of fraction of energy uncharged

Figure 5: Learning curves for the main experiment comparing the three MDPs, three values of L_c and investigating the impact of the RTC on performance

When comparing the influence of the value of L_c when no RTC is used, for $L_c = 15$ minutes MDP$_{\text{hid}}$ obtains the best performance as it shifts charging priority towards minimizing electricity cost rather than minimizing the fraction of uncharged energy as previously mentioned. For L_c=60 minutes with no RTC, the algorithm performs worse compared to the case for L_c=15 minutes with or without the RTC for MDP$_{\text{base}}$ and MDP$_{\text{agg}}$. This is due to the length of the control time step, which makes the control agent unable to learn the fluctuations in the PV electricity generation. Recall that, using the RTC allows to mitigate against the rapidly fluctuating PV generation especially when learning at a low time resolution. At $L_c = 5$ minutes, the high time resolution results in unstable learning for MDP$_{\text{base}}$ and MDP$_{\text{hid}}$. In MDP$_{\text{hid}}$, the total cost and the fraction of uncharged energy increase during training, while the electricity cost drops below that of the PIO. The RL algorithm hence has lost the ability to learn an effective policy. On the other hand, the lower dimension of the state-action space in MDP$_{\text{agg}}$ allows the RL algorithm to maintain its ability to learn (albeit slower than $L_c = 15$) an effective policy at $L_c = 5$ minutes, obtaining an electricity cost that is 0.1€ higher compared to that obtained when $L_c = 15$ minutes. It may be possible that the electricity cost decreases further for MDP$_{\text{agg}}$ and $L_c = 5$, possibly even dropping below that obtained for $L_c = 15$, after the measured 5×10^7 iterations. However, the CPU time required for testing this hypothesis would be impractical, with 5×10^7 iterations already requiring ≈ 24 hours.

The influence of the RTC is most noticeable at $L_c = 60$ minutes, resulting in a decrease in the electricity cost by on average 0.08€ ($\kappa = 1.5$) and 0.10€ ($\kappa = 5.0$) compared to the MDPs with $L_c = 60$ minutes without RTC. For $L_c = 15$ and $L_c = 5$ minutes the performance gain from the RTC is lower since those instances already learn on a high time resolution. The lowest electricity cost, $C^{elec} = 1.03$€, is obtained by MDP$_{\text{agg}}$ with RTC and $\kappa = 5.0$. This value is only 0.05€ above the electricity cost obtained by the PIO. Moreover, MDP$_{\text{agg}}$ with $L_c = 60$, with RTC and $\kappa = 5.0$ converges approximately two times faster and obtains an electricity cost of 0.1€ lower than MDP$_{\text{agg}}$ with $L_c = 15$ and without the RTC, which is the second best performing instance.

Figures 6 to 8 show examples of charging schedules for three sample days in the test set: a sunny day (Fig. 6), a day with variable sunshine (Fig. 7), and an overcast day (Fig. 8). The figures clearly show how the proposed framework moves the charging of EVs to later moments in the day when more PV-generated power is available and grid consumption prices are typically lower. This is a substantial improvement compared to the worst-case scenarios of the BAU strategy. The figures also show that, because of its prior knowledge, the PIO strategy is able to spread the actual charging over the entire time the EV is plugged in, resulting in less volatile charging power levels. It however does not always find the most optimal solution, as shown in Figure 7, where it is outperformed by the $\mathrm{MDP}_{\mathrm{agg}}$ that is able to avoid a small fraction of charging from the grid in the morning.

The behavior of the trained MDP instances on the sunny day is very similar, with all instances deferring charging to the middle of the day. With E^{req} and T^{dep} unknown to the agent in $\mathrm{MDP}_{\mathrm{hid}}$, it favours charging the EVs sooner - at a higher electricity cost - compared to $\mathrm{MDP}_{\mathrm{base}}$ to avoid an EV leaving with its battery not fully charged. However, as shown on the figure, EV_7 (gray) still leaves with an uncharged energy fraction of 0.117 due to its unusually short session around noon. The difference between $\mathrm{MDP}_{\mathrm{base}}$ and $\mathrm{MDP}_{\mathrm{agg}}$ is the order in which they charge the EVs. The heuristic dispatch in $\mathrm{MDP}_{\mathrm{agg}}$ prioritizes charging EV_2 (orange) in the morning since it has the lowest flexibility (due to its early departure time). In contrast, $\mathrm{MDP}_{\mathrm{base}}$ does not exhibit any logical charging priority.

On the test day with variable sunshine (Fig. 7), the benefit of using a smaller L_c or using the RTC is clearly visible. $\mathrm{MDP}_{\mathrm{agg}}$ with $L_c = 60$ and with the RTC obtains a perfect schedule ($C^{cons} = 0$) by being able to learn an effective policy on a broad timescale and adapting the charging power to the rapidly varying PV generation using the RTC. $\mathrm{MDP}_{\mathrm{agg}}$ without the RTC and $L_c = 5$ obtains a near perfect schedule. $\mathrm{MDP}_{\mathrm{hid}}$ fails to fully charge an EV with an unusually short session occurring around noon; EV_2 (yellow) leaves with an uncharged energy fraction of 0.212.

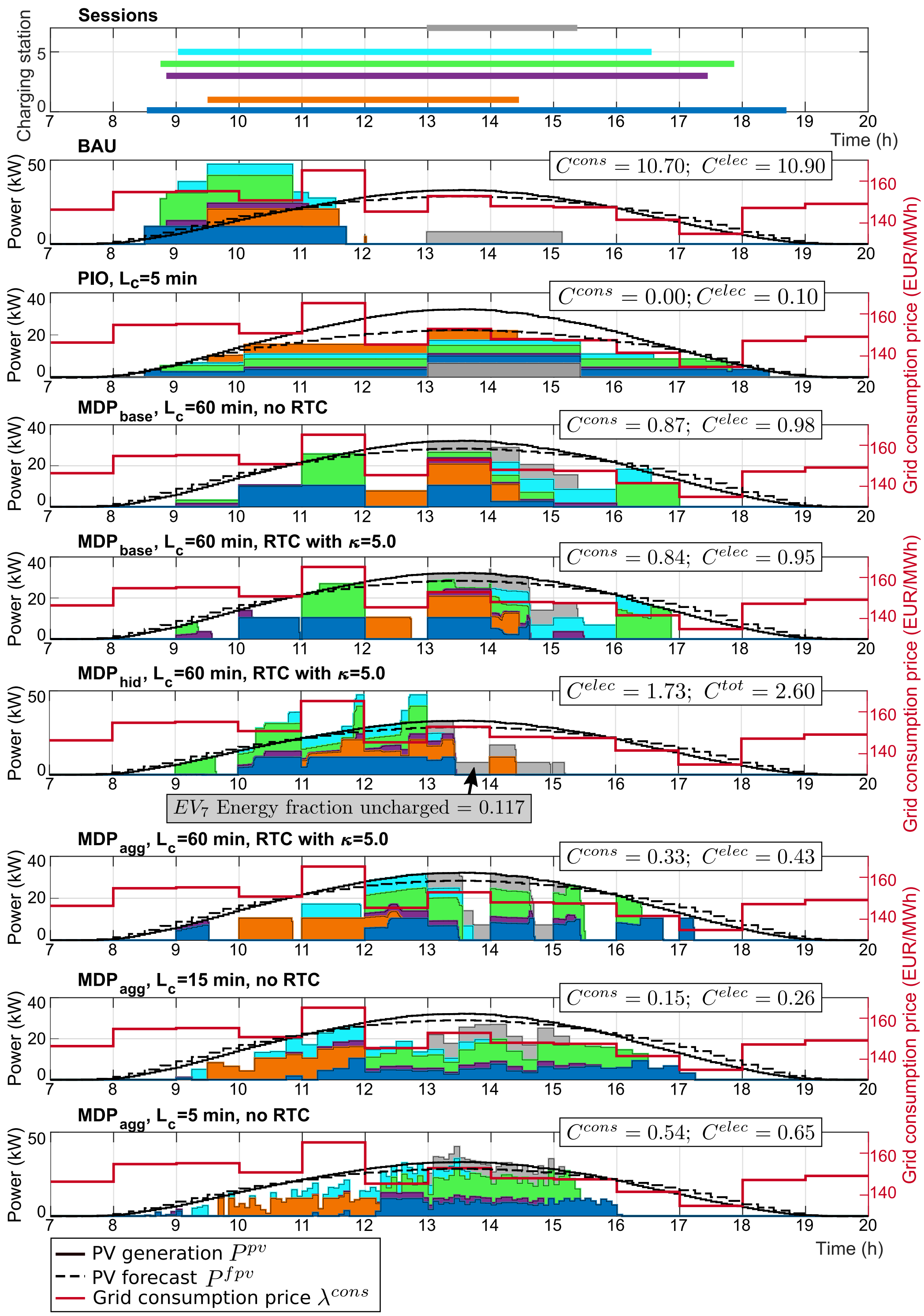


Figure 6: Timeline for a sunny test day for the BAU strategy and several instances of the RL algorithm

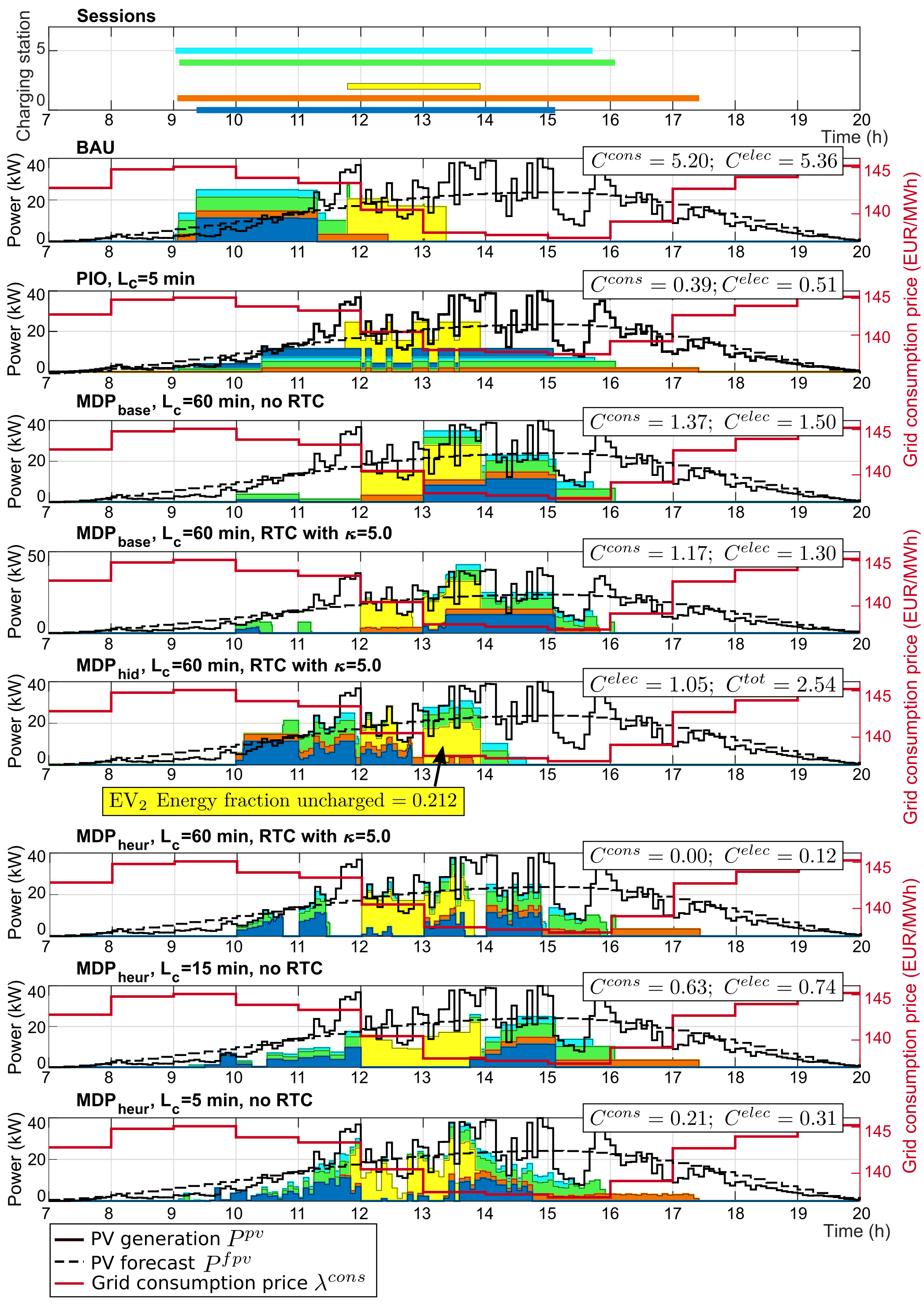


Figure 7: Timeline for a variable sunshine test day for the BAU strategy and several instances of the RL algorithm

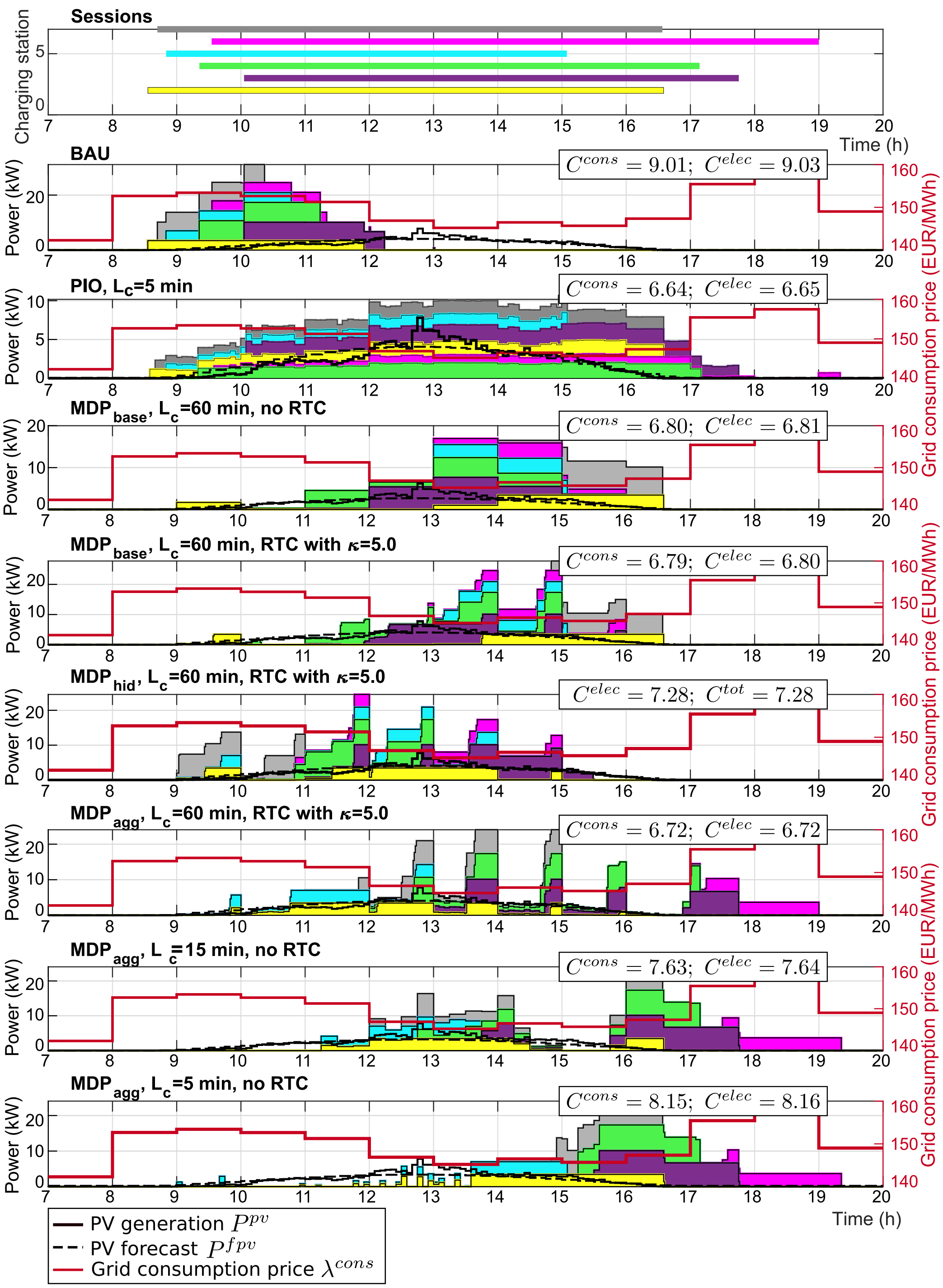


Figure 8: Timeline for an overcast test day for the BAU strategy and several instances of the RL algorithm

Finally, the optimal charging strategy on an overcast day (Fig. 8) would be to self-consume all of the PV generation and charge the remaining required energy during periods of low electricity prices. MDP_{agg} with $L_c = 60$ and with the RTC obtains the lowest electricity cost, self-consuming 100% of the PV-generation ($C^{inj} = 0$) and showing a trend towards charging the remaining energy during low λ^{cons}. The major downside of the RTC is that it is more volatile to EV charging power as can be seen in the figure. This volatility can be lowered by using a smaller value of κ, thereby trading the electricity cost for a slightly lower battery degradation. MDP_{hid} prioritizes charging earlier and this time manages to charge all EVs with their respective required energy. The instances with $L_c = 15$ and $L_c = 5$ minutes obtain the highest electricity costs, struggling to follow the PV generation and relying on the backup controller to charge the EVs at the end of their session.

The results from this experiment clearly show that the PPO algorithm can achieve an effective charging policy, and a careful design of the MDPs makes a significant difference in the performance of the algorithm. Combining the three-step method in MDP_{agg} and the RTC, results in a significantly lower electricity cost (0.2€ lower) compared to the straightforward design MDP_{base}. The resulting daily electricity cost is 1.03€, which is 1.71€ (62.5%) lower than the BAU strategy, and approaches the PIO within 0.05€ (5%). When E^{req} and T^{dep} are unknown, the choice of K_1 must be done carefully such that it results in the desired trade-off between electricity cost and the amount of uncharged energy. For example, if for 2% of the sessions, the EV leaves with a fraction of uncharged energy greater than 25%, a daily electricity cost improvement over BAU of 1.31€ (48%) is obtained. This value of the electricity cost is 0.4€ (39%) higher compared to the best performing instance when the departure time and energy require to fully charge the EV are known.

4.2.2. Experiment 2: scalability in terms of fleet size

This experiment evaluates the scalability of the proposed control framework to larger fleet sizes for the three MDPs. The historical data at EnergyVille containing measurements for $N_{ev} = 8$, is used to generate training and testing data for other fleet sizes. The BAU strategy, PIO strategy and the RL algorithm with the three MDPs are tested for $N_{ev} = \{2, 8, 16, 32\}$. The MDP hyper-parameters are $L_c = 60$, $N_{past} = N_{fut} = 2$ and $N_{par} = 20 \times N_{ev}/8$ (MDP_{agg}). For $N_{ev} = 2$, the actor-critic network is modified to one shared layer with 64 nodes and two layers with 32 nodes each for the actor and the critic networks. Additionally, for $N_{ev} = 2$, the `learning_rate` PPO hyper-parameter is set to 0.0004 (four times higher than before). Due to the limited computational resources, the PIO strategy is trained with $L_c = \{15, 15, 30, 60\}$ [minutes] for $N_{ev} = \{2, 8, 16, 32\}$.

The resulting learning curves are shown in Fig. 9 and numerical results, shown in Table 1, contain for each instance the mean value of the electricity cost between time steps 9.6×10^6 and 14.4×10^6. The RL algorithm learns an effective policy for all three MDPs and for all tested fleet sizes with a reduction in electricity cost between 46% and 63%.

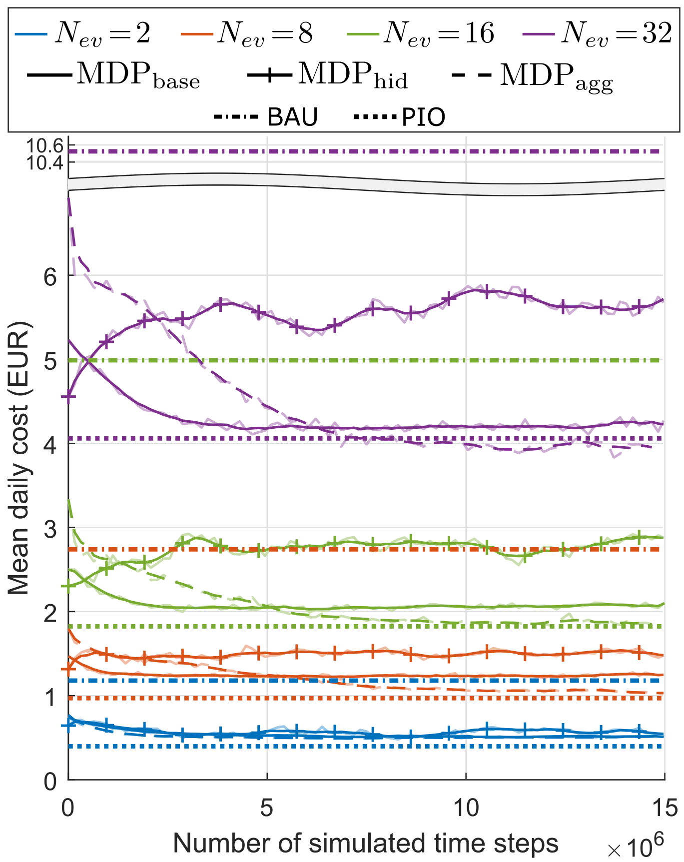


Figure 9: Mean daily C^{elec}[€] measured on test set for BAU, PIO and the three MDPs for several simulated fleet sizes N_{ev}

Compared to $\mathrm{MDP_{agg}}$, $\mathrm{MDP_{base}}$ converges faster but obtains a higher final cost. At $N_{ev} = 2$ both MDPs obtain a similar performance. For larger fleet sizes, the absolute difference in electricity cost between the MDPs increases,

$N_{ev} =$	2	8	16	32
$\mathrm{MDP}_{\mathrm{base}}$	0.515	1.238	2.058	4.208
$\mathrm{MDP}_{\mathrm{hid}}$	0.583	1.506	2.740	5.698
$\mathrm{MDP}_{\mathrm{agg}}$	0.505	1.051	1.871	3.968
BAU	1.179	2.739	4.988	10.525
PIO	0.398	0.972	1.824	4.059

Table 1: Mean daily C^{elec}[€] measured on test set for BAU, PIO, and the three MDPs for several simulated fleet sizes N_{ev}

for example a difference of 0.187€ for $N_{ev} = 8$ and 0.240€ for $N_{ev} = 32$. The curse of dimensionality makes learning more difficult in high dimensional state-action spaces, and is noticeable for $\mathrm{MDP}_{\mathrm{base}}$. Due to the state-action space aggregation, $\mathrm{MDP}_{\mathrm{agg}}$ is able to mitigate the curse of dimensionality. At $N_{ev} = 32$, the electricity cost for $\mathrm{MDP}_{\mathrm{base}}$ is less than that obtained by the PIO. This is mainly due to the ability to take an aggregate action for the whole fleet rather than individual actions making it more computationally feasible.

The simulation results presented above show that the proposed RL control framework is suitable for coordinating the charging of a fleet of EVs. When knowledge on the departure time of the EV and the energy required to fully charge the EV before its departure are available, the proposed control framework has limited scalability issues especially when the aggregate MDP formulation is used. Even though the proposed control framework outperforms the BAU model when knowledge on departure time and energy required to fully charge the EV are not available, there is no guarantee of the EV being fully charged before departure. Furthermore, it is worth mentioning that scalability is expected beyond the tested fleet sizes, especially when MDP_{agg} is used as has been shown in [23].

In summary, to learn a cost effective control policy to efficiently coordinate the charging of large EV fleets, it is necessary to invest in a charging infrastructure that allows obtaining information on the departure time of the EVs and the energy required to fully charge the EVs before departure. The proposed control framework with the aggregate MDP, the real-time controller, and a control time step of 60 minutes would be a suitable choice for coordinated charging of large EV fleets.

5. Conclusion

This paper proposes a hybrid control framework that combines the strengths of reinforcement learning and rule-based control for coordinating the charging of an electric vehicle fleet in an office building. Specifically, the framework applies proximal policy optimization, a policy-gradient based reinforcement learning algorithm, to provide a charging schedule with coarse time granularity, which is refined by a rule-based controller to per minute real-time control actions. The control objective is to maximise self-consumption of the local electricity generation and minimize electricity cost. The performance of the proposed framework was evaluated using real-world data from an office building.

The simulation results show that the proposed control framework successfully schedules the charging of an electric vehicle fleet to achieve the control objective. It largely outperforms a business-as-usual strategy and approaches a near-optimal strategy with a 5% performance gap when charging sessions are aggregated before optimization. Simulation results equally show performance improvements when information on departure time and required energy is available, and when real-time information on local photo-voltaic electricity generation is used to optimize on a fine time scale.

Future work aims to investigate hierarchical reinforcement learning as a replacement for the proposed hybrid method. A downside of the proposed rule-based real-time controller is that it requires expert knowledge and is not generalisable to other settings with a different objective function, such as peak shaving. Hierarchical reinforcement learning may provide a generalisable and scalable solution for learning on both coarse and fine timescales. Additionally, the proposed framework will be evaluated, and where needed adapted, to better serve the existing range of charging modalities. Specifically the support of fast-charging and vehicle-to-grid charging is of interest to better support the available charging infrastructure at EnergyVille.

Acknowledgement

This research is funded by the Flemish Institute for Technological Research (VITO) through a PhD scholarship, and the ROLECS project – Flux50-VLAIO-HBC.2018.0527.

References

[1] International Energy Agency, Global EV Outlook 2019, https://www.iea.org/reports/global-ev-outlook-2019, 2019. Accessed: 2020-08-07.

[2] M. Blonsky, A. Nagarajan, S. Ghosh, K. McKenna, S. Veda, B. Kroposki, Potential impacts of transportation and building electrification on the grid: A review of electrification projections and their effects on grid infrastructure, operation, and planning, Current Sustainable/Renewable Energy Reports 6 (2019) 169–176. doi:10.1007/s40518-019-00140-5.

[3] J. Hu, S. You, M. Lind, J. Østergaard, Coordinated charging of electric vehicles for congestion prevention in the distribution grid, IEEE Transactions on Smart Grid 5 (2013) 703–711. doi:10.1109/TSG.2013.2279007.

[4] M. Van Der Kam, W. van Sark, Smart charging of electric vehicles with photovoltaic power and vehicle-to-grid technology in a microgrid; a case study, Applied energy 152 (2015) 20–30. doi:10.1016/j.apenergy.2015.04.092.

[5] N. Sadeghianpourhamami, J. Deleu, C. Develder, Definition and evaluation of model-free coordination of electrical vehicle charging with reinforcement learning, IEEE Transactions on Smart Grid 11 (2020) 203–214. doi:10.1109/TSG.2019.2920320.

[6] A. R. Bhatti, Z. Salam, A rule-based energy management scheme for uninterrupted electric vehicles charging at constant price using photovoltaic-grid system, Renewable energy 125 (2018) 384–400. doi:10.1016/j.renene.2018.02.126.

[7] D. Wang, F. Locment, M. Sechilariu, Modelling, simulation, and management strategy of an electric vehicle charging station based on a DC microgrid, Applied Sciences 10 (2020) 2053. doi:https://doi.org/10.3390/app10062053.

[8] A. Di Giorgio, F. Liberati, S. Canale, Electric vehicles charging control in a smart grid: A model predictive control approach, Control Engineering Practice 22 (2014) 147–162. doi:10.1016/j.conengprac.2013.10.005.

[9] S. Bansal, M. N. Zeilinger, C. J. Tomlin, Plug-and-play model predictive control for electric vehicle charging and voltage control in smart grids, in: 53rd IEEE Conference on Decision and Control, IEEE, 2014, pp. 5894–5900. doi:10.1109/CDC.2014.7040312.

[10] B.-R. Choi, W.-P. Lee, D.-J. Won, Optimal charging strategy based on model predictive control in electric vehicle parking lots considering voltage stability, Energies 11 (2018) 1812. doi:10.3390/en11071812.

[11] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

[12] J. R. Vázquez-Canteli, Z. Nagy, Reinforcement learning for demand response: A review of algorithms and modeling techniques, Applied Energy 235 (2019) 1072 – 1089. doi:10.1016/j.apenergy.2018.11.002.

[13] S. Vandael, B. Claessens, D. Ernst, T. Holvoet, G. Deconinck, Reinforcement learning of heuristic EV fleet charging in a day-ahead electricity market, IEEE Transactions on Smart Grid 6 (2015) 1795–1805. doi:10.1109/TSG.2015.2393059.

[14] A. Chiş, J. Lundén, V. Koivunen, Reinforcement learning-based plug-in electric vehicle charging with forecasted price, IEEE Transactions on Vehicular Technology 66 (2016) 3674–3684. doi:10.1109/TVT.2016.2603536.

[15] S. Wang, S. Bi, Y. A. Zhang, A reinforcement learning approach for EV charging station dynamic pricing and scheduling control, in: 2018 IEEE Power Energy Society General Meeting (PESGM), 2018, pp. 1–5. doi:10.1109/PESGM.2018.8586075.

[16] Z. Wan, H. Li, H. He, D. Prokhorov, Model-free real-time EV charging scheduling based on deep reinforcement learning, IEEE Transactions on Smart Grid 10 (2019) 5246–5257. doi:10.1109/TSG.2018.2879572.

[17] J. Lee, E. Lee, J. Kim, Electric vehicle charging and discharging algorithm based on reinforcement learning with data-driven approach in dynamic pricing scheme, Energies 13 (2020) 1950. doi:10.3390/en13081950.

[18] C. J. C. H. Watkins, Learning from delayed rewards (1989). King's College, Cambridge.

[19] L. Yu, W. Xie, D. Xie, Y. Zou, D. Zhang, Z. Sun, et al., Deep Reinforcement Learning for Smart Home Energy Management, IEEE Internet of Things Journal 7 (2020) 2751–2762. doi:10.1109/JIOT.2019.2957289.

[20] D. Qiu, Y. Ye, D. Papadaskalopoulos, G. Strbac, A deep reinforcement learning method for pricing electric vehicles with discrete charging levels, IEEE Transactions on Industry Applications 56 (2020) 5901–5912. doi:10.1109/TIA.2020.2984614.

[21] H. Li, Z. Wan, H. He, Real-time residential demand response, IEEE Transactions on Smart Grid 11 (2020) 4144–4154. doi:10.1109/TSG.2020.2978061.

[22] A. Moonens, A. Nowé, Fine-grained control of electric vehicle charging with policy gradient, in: Proceedings of the Adaptive and Learning Agents Workshop 2019 (ALA-19), 2019.

[23] S. Vandael, B. Claessens, M. Hommelberg, T. Holvoet, G. Deconinck, A scalable three-step approach for demand side management of plug-in hybrid vehicles, IEEE Transactions on Smart Grid 4 (2013) 720–728. doi:10.1109/TSG.2012.2213847.

[24] P. E. Gill, E. Wong, Sequential quadratic programming methods, in: Mixed integer nonlinear programming, Springer, 2012, pp. 147–224.

[25] Z. Wan, H. Li, H. He, D. Prokhorov, Model-Free Real-Time EV Charging Scheduling Based on Deep Reinforcement Learning, IEEE Transactions on Smart Grid 10 (2019) 5246–5257. doi:10.1109/TSG.2018.2879572.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization Algorithms, CoRR abs/1707.06347 (2017). URL: http://arxiv.org/abs/1707.06347.

[27] D. Kraft, A software package for sequential quadratic programming, Ein Software-Paket zur sequentiellen quadratischen Optimierung, Forschungsbericht., Technical Report, Institut für Dynamik der Flugsysteme, Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt $DFVLR$, Oberpfaffenhofen, Braunschweig, 1988.