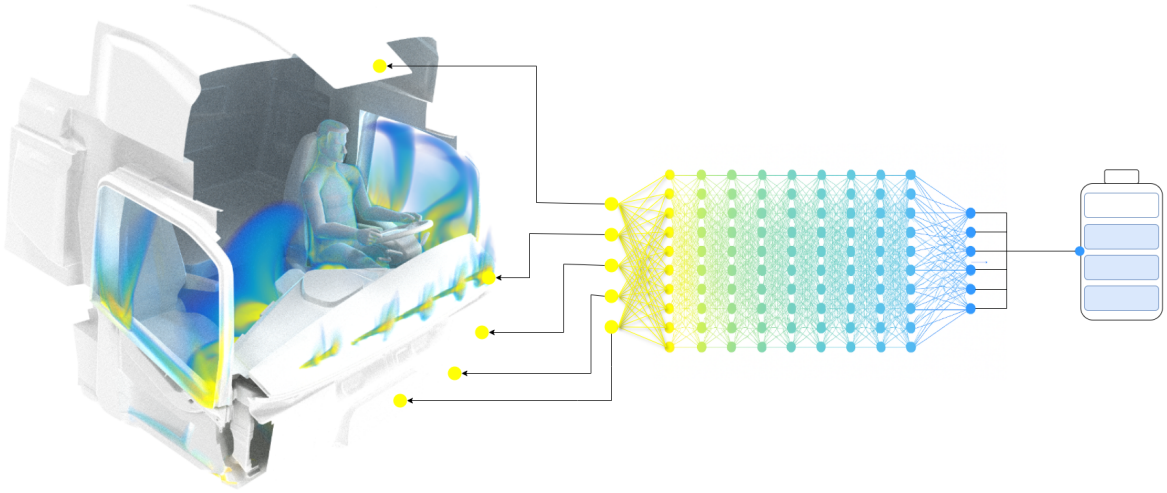
CHALMERS
UNIVERSITY OF TECHNOLOGY



Adaptive Cabin Climate Control for Battery Electric Vehicles Using TD3 Reinforcement Learning

Master's Thesis in Complex Adaptive Systems

ABUBAKAR ABUKAR

DEPARTMENT OF ARCHITECTURE AND CIVIL ENGINEERING
Division of Geology and Geotechnics

CHALMERS UNIVERSITY OF TECHNOLOGY
Master's Thesis ACEX30
Gothenburg, Sweden 2025

MASTER'S THESIS ACEX30

Adaptive Cabin Climate Control for Battery Electric Vehicles Using TD3 Reinforcement Learning

Master's Thesis in the Master's Programme Complex Adaptive Systems

ABUBAKAR ABUKAR

Department of Architecture and Civil Engineering
Division of Geology and Geotechnics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025

Adaptive Cabin Climate Control for Battery Electric Vehicles Using TD3 Reinforcement Learning
Master's Thesis in the Master's Programme Complex Adaptive Systems

ABUBAKAR ABUKAR

Examensarbete ACEX30
Institutionen för Arkitektur och Samhällsbyggnadsteknik
Chalmers Tekniska Högskola, 2025

Department of Architecture and Civil Engineering
Division of Geology and Geotechnics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone +46 31 772 1000

Department of Architecture and Civil Engineering
Göteborg, Sweden, 2025

Acknowledgements

All Praise belongs to the Most High.

This thesis has come to fruition through the guidance of many people, most notably my supervisors, Dr. Jelena Andric, Maria Krantz, and Chaithanya Ramkumar. Dr. Jelena Andric has been an immense figure for both facilitating the possibility of me performing this thesis, as well as aiding in formulating the right research question throughout the journey. I would also like to thank Maria Krantz for her continuous support and coordination throughout the project, which greatly contributed to its progress and success. Chaithanya Ramkumar's technical expertise has been invaluable in this work. His vision laid the foundation for this project and greatly shaped its development.

I am thankful to my examiner, Dr. Kun Gao, for his trust and for allowing me the creative freedom to explore and define the direction of this thesis.

I am enormously grateful to all my colleagues in the Thermal Management department of Volvo Group Truck Technology for generously sharing their expertise and for continually encouraging my curiosity.

Adaptive Cabin Climate Control for Battery Electric Vehicles Using TD3 Reinforcement Learning

Master's thesis in the Master's Programme Complex Adaptive Systems

ABUBAKAR ABUKAR

Department of Architecture and Civil Engineering
Division of Geology and Geotechnics
Chalmers University of Technology

ABSTRACT

The transition from combustion engine vehicles to Battery Electric Vehicles (BEVs) has increased the importance of efficient thermal management in trucks. Conventional cabin climate control methods prioritize transparency and safety but lack adaptability, limiting potential energy savings. This thesis addresses this challenge using the Twin-Delayed Deep Deterministic Policy Gradient (TD3) combined with a novel surrogate model, the Twin Fourier Neural Operator (Twin FNO), designed to capture cabin thermodynamics through partially enforced physics and dual output heads for improved accuracy. The Twin FNO predicts average cabin temperature with a mean absolute error of 0.55 °C while being 180 times faster than numerical solvers. Integrated into the TD3 framework, the agent effectively balances energy efficiency and thermal comfort. It maintains the setpoint temperature when energy is abundant and deliberately creates an offset when energy is limited, achieving up to 40% reduction in Heating Ventilation and Air Conditioning (HVAC) energy consumption with a 6 °C temperature deviation. These results highlight the potential of reinforcement learning and surrogate modeling to enable energy-adaptive thermal control strategies in BEVs while raising questions on acceptable thermal comfort trade-offs.

Keywords: Battery Electric Vehicles, Thermal Management, Control Strategies, Reinforcement Learning, TD3, Fourier Neural Operator

Adaptiv Kabin Klimat Kontrol för Batteri Elektriska Fordon genom TD3 Reinforcement Learning

Examensarbete inom masterprogrammet Komplexa Adaptiva System

ABUBAKAR ABUKAR

Institutionen för arkitektur och samhällsbyggnadsteknik
Avdelningen för Geologi och Geoteknik
Chalmers tekniska högskola

SAMMANFATTNING

Övergången från förbränningsmotors fordon till batterielektriska fordon (BEV) har ökat betydelsen av effektiv termisk styrning i lastbilar. Konventionella metoder för klimatkontroll i hytter prioriterar transparens och säkerhet men saknar anpassningsförmåga, vilket begränsar potentiella energibesparingar. Detta examensarbete adresserar denna utmaning genom att använda Twin-Delayed Deep Deterministic Policy Gradient (TD3) i kombination med en ny surrogatmodell, Twin Fourier Neural Operator (Twin FNO), som är utformad för att fånga hyttens termodynamik med delvis införda fysikaliska lagar och dubbla utgånger för förbättrad noggrannhet. Twin FNO förutsäger den genomsnittliga hyttemperaturen med ett medelfel på 0,55 °C och är 180 gånger snabbare än numeriska lösare. Integrerad i TD3-ramverket balanserar agenten effektivt energieffektivitet och termisk komfort. Den upprätthåller börvärdestemperaturen när energi är rikligt tillgänglig och tillåter avvikelser när energitillgången är begränsad, vilket uppnår upp till 40% minskning av Heating, Ventilation and Air Conditioning (HVAC) energi konsumption med en temperaturavvikelse på 6 °C. Dessa resultat visar potentialen hos förstärkningsinlärning och surrogatmodellering för att möjliggöra energiadaptiva termiska styrstrategier i BEV, samtidigt som de väcker frågor kring acceptabla kompromisser för termisk komfort.

Nyckelord: Batterielektriska fordon, Termisk hantering, Reglerteknik, Reinforcement Learning, TD3, Fourier Neural Operator

Table of Contents

List of Figures

List of Abbreviations

AI Artificial Intelligence

BEV Battery Electric Vehicles

CC Cabin Climate

COP Coefficient of Performance

DDPG Deep Deterministic Policy Gradient

DQN Deep Q-Network

FNO Fourier Neural Operator

HVAC Heating, Ventilation and Air Conditioning

MAE Mean Absolute Error

MDP Markov Decision Process

ML Machine Learning

MSE Mean Squared Error

PI Proportional-Integral

PID Proportional-Integral-Derivative

PINN Physics-Informed Neural Network

PPO Proximal Policy Optimization

RL Reinforcement Learning

SAC Soft Actor-Critic

SoC State of Charge

TD3 Twin Delayed Deep Deterministic Policy Gradient

Chapter 1

Introduction

The transportation industry is undergoing a rapid transformation with the widespread adoption of Battery Electric Vehicles (BEV). Among the various design challenges for BEV, energy efficiency remains a critical focus, particularly in the context of climate control systems. Unlike internal combustion engine vehicles, where waste heat can be utilized for cabin heating, BEV rely on their limited battery capacity to power thermal regulators. This introduces a trade-off between thermal efficiency and driving range, making efficient climate control systems crucial for extending vehicle range and improving overall thermal comfort.

A large amount of energy is reserved for Cabin Climate (CC) control. According to fleet data, between 14% and 18% [3] of the total State of Charge (SoC) is used for thermal regulation, which include controlling heaters, evaporators and air flow to the cabin. In order to optimize range, it is therefore vital to investigate strategies for efficient utilization of the energy allocated for thermal control.

Current control strategies deployed in most vehicles are dominated by simple feedforward or feedback loop controllers such as Proportional-Integral (PI) and Proportional-Integral-Derivative (PID) controllers. These models are often aided by rule-based logic as well as sensors. The controllers work by regulating actuators based on the difference between current sensor-relayed values of the control variables (often temperature) and the setpoint. The appeal in these strategies lies in their inherent safety and simplicity. PID gains have been deployed and refined for decades and as such are well understood. With fairly simple calibration, these models can effectively reach and maintain the setpoint temperature, and as such have been the industry standard for some time.

However, the same inherent simplicity is also their limitation in more complex scenarios. While conventional controllers are effective for fulfilling the objective of thermal comfort, balancing this objective with energy efficiency is a tall task for simple models. Optimizing contradictory objectives requires adaptability which is not possible for PID loops, because they regulate the actuators based solely on temperature error. Other vital factors for energy efficiency, such as SoC and projected range, are not present in the decision making, so the only way to balance additional objectives is through impromptu

rule-based logic and heuristics.

Recent advancements in Artificial Intelligence (AI) have opened up new opportunities to optimize complex systems. Reinforcement Learning (RL), a subset of Machine Learning (ML), has demonstrated remarkable success in dynamically controlling systems by learning optimal policies that optimize multiple objectives through a reward system. For BEV cabin climate control, RL offers the potential to balance comfort and energy efficiency by learning to micro-control thermal regulators in response to both external and internal factors, such as ambient temperature and available energy. RL is therefore inherently a valuable alternative that solves the shortcomings of conventional control strategies in complex adaptive systems. This thesis will cover the work done to develop and evaluate the application of deep RL as a controller that learns to regulate cabin temperature to optimize both thermal comfort and energy efficiency in BEV.

1.1 Problem Formulation

The aim of this thesis is to leverage reinforcement learning to train an adaptive agent to control thermal actuators, specifically heater, evaporator and blower, in a BEV truck in order to efficiently regulate cabin temperature and thereby increase total range. Achieving this goal involves addressing two key challenges:

- **Train an agent** that balances passenger comfort and energy consumption to find an optimal policy that meets the desired requirements. The main challenge is to define a reward function that will converge on an optimal policy while inherently being adaptive.
- **Train a surrogate model** to accurately mimic the responses of the environment. The surrogate model must capture the underlying physics that govern thermodynamics in the cabin.

In order to make valuable strides in this research, certain limitations are unavoidable. Firstly, the term of thermal comfort will be used frequently throughout this work. Thermal comfort is concept that encompass many variables and aspects, however in this work it will only refer to average cabin temperature. Secondly, due to time constraints this stage of the project will only consider soak temperature equal to the ambient temperature. This means the initial temperature of the cabin is equal to the surrounding air. The framework and approach will be almost identical without these constraint and so this work will still be valuable.

Chapter 2

Cabin Climatization

This chapter lays the technical groundwork for the control task. Section 2.1 details the cabin's thermal dynamics, how key thermal regulators govern airflow and temperature.

2.1 Thermodynamics of the Cabin

2.1.1 Thermal Loads

In order to properly understand the context it is necessary to understand the underlying thermal dynamics of the system the agent is tasked to control. Vehicle cabins are often viewed as lumped models as the alternative are computationally heavy, so much so that it would be infeasible to conduct this study. For this reason, this project will use a simplified one-dimensional lumped model. The heat equation in this case is the sum of all relevant heat transfers that occur in a cabin. Previous work has shown that these can be split into external thermal interactions as well as the internal interactions [2]. External interactions include interactions between the cabin and the sun, surrounding air, other vehicle components as well as unavoidable losses due to leakage. The internal interactions on the other hand depend on objects and passengers within the cabin itself. These are modeled as separate parts since the model targets only the air temperature. Examples of the internal thermal interactions are the natural expelling of heat from the human body. As our internal temperature is often much higher than the temperature of the cabin, convection occurs. The temperature of the objects within the cabin also affects the air temperature since each material has a unique thermal mass that absorbs and expels heat at different rates compared to the air.

The thermal load is the Heating, Ventilation and Air Conditioning (HVAC) energy consumption, which is left out of both groups as this load is the entirely manufactured exchange between the cabin and the surroundings. This load is the controlled addition, or subtraction depending on the situation, that occurs through the air intake. How exactly this term is controlled is described in the following section. While the HVAC load should logically be determined in accordance to the remaining loads in order to reach a desired equilibrium, most control theories determine this load by observing the changes in temperature. This work will use the same approach, therefore, while all loads are relevant for modeling the system, this work is focused solely on the HVAC load.

2.1.2 HVAC System Components

In essence, the HVAC load is regulated through airflow from the surroundings to the cabin, which is heated or cooled depending on the situation. The HVAC system, depicted in figure 2.1, consists of a number of components that interact with the air in order. The air is first sucked into the blower that controls the total mass flow that enters the system, and thereby eventually enters the cabin. All of the air flows from the blower into the evaporator where the level of cooling is controlled by the refrigerant loop. After the evaporator the airflow is controlled by a bypass valve which controls the amount of the air flow that can bypass the heater, and the amount that airflow that enters the heater before mixing with the remaining air. After mixing, the air enters the cabin after which it can be recirculated and reenter the system before the blower. Recirculation is done in order to increase energy efficiency but comes with risks.

The refrigerant loop that allows the evaporator to remove heat from the air flow is a vapor compression loop in which the heat is moved from the evaporator to the condenser. This is made possible by pressurizing the coolant before entering the evaporator and depressurizing it before it enters the condenser. Essentially, the loop pays for the cost of moving the heat with compression work, making the compressor the energy consumer. However, it is the need for cooling by the evaporator that is used as a control signal for the compressor. This fact allows us to view the evaporator as a standalone component that removes heat based on the need, and not based on the loop itself. For the sake of simplicity, the refrigerant loop as a whole will therefore not be considered in this work.

Additionally, the airflow is further controlled by a system of vents and fans in order to further control how it enters the cabin, however that will be treated as a separate system that is not covered in this work. While it is a vital system for thermal comfort, this work focuses on the temperature and mass flow of the air that is regulated by the HVAC system.

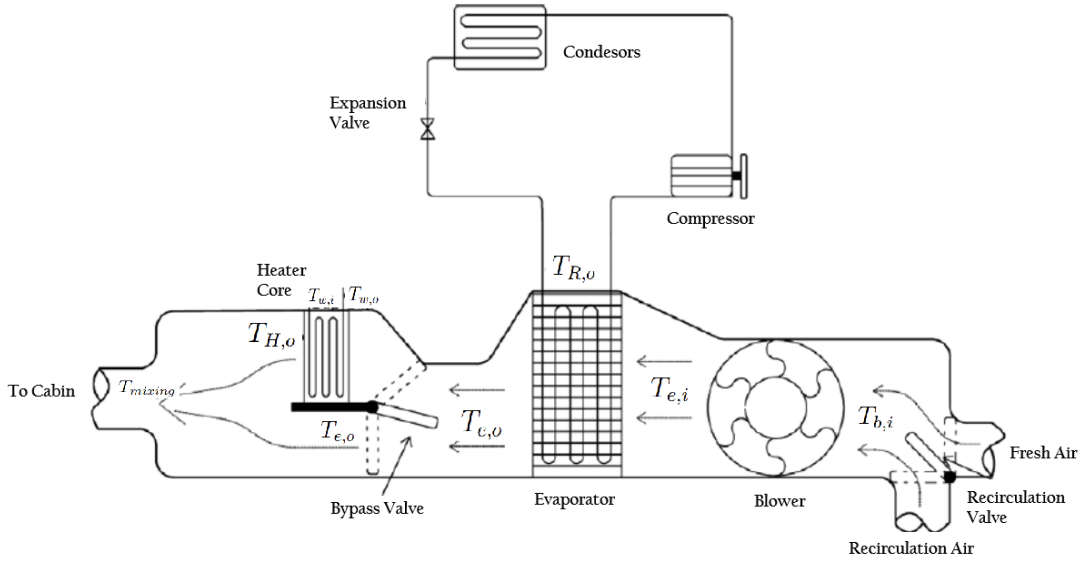


Figure 2.1: Schematic of the BEV cabin climate circuit. Ambient air enters at the front intake, the blower sets mass-flow, the evaporator provides cooling, a blend door diverts part of the flow through the heater, and the mixed stream is discharged into the cabin. Adapted from [2]

2.1.3 HVAC Control Strategies

While certain operating modes are obvious, heating mode in cold conditions and cooling mode in warm conditions, certain caveats are important to note in well developed control strategies. Firstly, the heating mode here refers to turning the evaporator off and heating the air, thereby adding heat to the cabin. Cooling mode refers to the opposite, subtracting heat from the cabin by injecting cool air. However, there are situations where both cooling and heating the same air is advantageous. This is due to the fact that warm air in cold conditions runs the risk of fogging the window, preventing the driver from seeing the road. Due to this safety risk, energy is spent to cool the air in the evaporator, thereby reducing the humidity in the air. The fog risk in heating mode is also the reason why recirculating the air is risky, as the humidity will accumulate.

At the opposite end of the spectrum the fog risk is significantly lower. This allows for the possibility of recirculating the air, effectively reducing the energy demand on the cooler. The problem in this case is that continuous recirculation leads to accumulating CO_2 content in the air. To combat this, some fresh air is necessary. The mix of cool, recirculated air and warm fresh air may reach temperatures below the setpoint temperature, making heating necessary.

Chapter 3

Literature Study

In recent years, a variety of RL algorithms have been applied to building HVAC control, ranging from early value-based methods to modern policy-gradient techniques. Early work often used discrete-action Deep Q-Network (DQN) or tabular Q-learning to learn HVAC scheduling policies. For example, Wei et al. (2017) trained a DQN agent in EnergyPlus simulations to optimally pre-cool or pre-heat a building, taking into account time-of-use electricity pricing [10]. That DQN-based controller achieved substantial cost savings (up to 20-70% energy cost reduction compared to a rule-based baseline) while maintaining indoor temperatures within comfort bounds. However, HVAC control inherently involves continuous control variables, and using discrete approximation can limit performance. Accordingly, more recent studies [9] have favored continuous-action RL algorithms such as Deep Deterministic Policy Gradient (DDPG) and its improved successor Twin Delayed Deep Deterministic Policy Gradient (TD3), as well as stochastic policy methods like Proximal Policy Optimization (PPO) and Soft Actor-Critic (SAC). These algorithms can directly handle continuous control inputs and have been shown to improve learning stability and control performance in HVAC tasks. For instance, TD3 has been used in several building HVAC studies as a stable off-policy learner, while SAC has recently emerged as a strong alternative with sample-efficient learning and robust convergence [9]. As one comparative review notes, the field is moving beyond DQN to such modern RL algorithms, which better capture the continuous, multi-variable nature of HVAC control [1] [9]. Overall, the consensus in recent literature is that advanced deep RL frameworks (DDPG/TD3, PPO, SAC) tend to outperform simple DQN or heuristic controllers for HVAC, by learning more nuanced control policies that account for the building's continuous dynamics.

DeepMind's 2016 data-center project [8] showed RL can handle complex HVAC control at scale, cutting cooling energy by approximately 40% (15% of total facility power). Follow-up field trials in two commercial buildings [8] trimmed HVAC energy by 9% and 13% over tuned industrial baselines while meeting safety and comfort constraints. Zhang et al. (2019) transferred an actor-critic policy from simulation to a real office radiant-heating system, saving approximately 16% heating energy. Numerous benchmarks [1] [8] [9] show modern RL algorithms (TD3, SAC, PPO) reliably beat model-predictive or rule-based controls by $10 - 30\%$ while respecting comfort limits, confirming RL's practical value for building HVAC.

Chapter 4

Methodology

In the sections that follow, we first define the framework of RL, then define the environment and finally detail the TD3 implementation.

4.1 Reinforcement Learning

RL is a field that has been studied and developed for quite some time and thereby has many different versions. However, these different algorithms are based on the same principles as are described in this section.

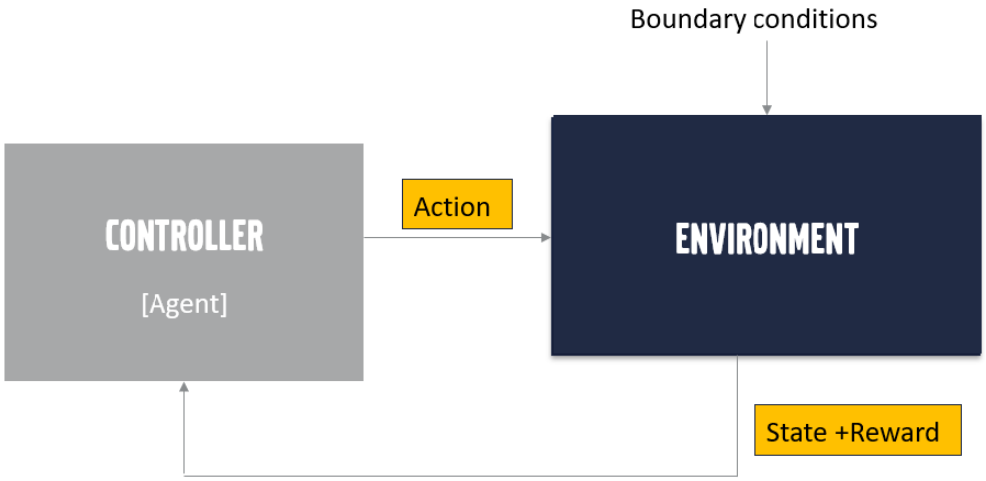


Figure 4.1: RL control loop cast as an MDP. At each step the agent observes the state vector, chooses an action (blower, evaporator, heater power), the environment transitions a new state, and a composite reward for comfort and energy is returned along with the new state.

The central part of the RL framework is a Markov Decision Process (MDP) in which the environment is modeled as a tuple

$$\big(\mathcal{S},\ \mathcal{A},\ P,\ R\big),$$

where $\mathcal{S}$ is a set of all possible states, $\mathcal{A}$ is a set of all possible actions, $P(s_{t+1} \mid s_t, a_t)$ is the transition dynamics and $R(s_t, a_t)$ is the reward function. In a Markov Decision Process (MDP), the objective is to control the environment such that the accumulated rewards are maximized. RL is the process of letting an agent/actor explore the search space to find an optimal policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that fulfills the objective of MDP through an iterative process of continued interaction with the environment. The interactions consist of the agent observing the state of the environment s_t, the agent then performs an action a_t on the environment, after which the environment transitions and returns a new state s_{t+1} as well as a reward r_t for the previous action. The purpose of the reward is to quantify how good the action was given the state of the environment when the action was performed. This process is done in a loop as depicted in figure 4.1. In essence, the agent learns from previous experiences, adapts the policy and then tries again, in a loop, until the policy is fully optimized. Experiences are saved in a tuple of state, action, following state and reward which are used for training.

This framework is the basis on which RL has been developed. RL as a subset of ML is divided into many algorithms that use this framework but vary in both complexity and utility. At one end are value-based approaches such as Q-learning or its deep learning variant DQN, which learn a state–action value table or network and suit discrete, low-dimensional action spaces. Policy-gradient methods (e.g., REINFORCE, PPO) learn a stochastic policy directly, providing smoother control at the cost of higher sample demand. Bridging the two are actor–critic frameworks, notably DDPG, TD3, and SAC. These pair a policy (actor) with a value estimator (critic) and can act in high-dimensional, continuous spaces. Actor–critic methods further split into off-policy learners that recycle past experience via replay buffers (DDPG, TD3, SAC) and on-policy learners that rely only on fresh rollouts (PPO, A2C). This spectrum of algorithms supplies a toolbox from which the appropriate algorithm can be matched to the task at hand. In this case, the TD3 algorithm was identified as the best algorithm for controlling HVAC for a number of reasons. First of which is that our problem involves continuous state and action space. Secondly, the virtual environment used in this study provides us with the option of storing experiences in a replay buffer opening up the door for off-policy learners. Among the off-policy learners TD3 was picked over SAC for being deterministic, which is vital for safety reasons. Both DDPG and TD3 have shown to be powerful frameworks but recent research have shown TD3 to be superior in most cases [4] and for that reason the TD3 algorithm has been used in this work.

4.2 RL Algorithm

To understand the TD3 algorithm and all its parts, it is easier to first understand the DDPG as it is the predecessor to TD3.

4.2.1 DDPG

The challenge with continuous state and action spaces is related to the original implementations of RL. Early algorithms use a table for storing information about the quality of each action, with the size of the table being determined by all possible states and actions. With an increasingly large number of possibilities this table then quickly grows to an untenable size. Deep RL algorithms were first introduced to tackle this issue by using a deep neural network from which the quality could simply be inferred from, thereby

bypassing the need of storing in a rigid table. The Critic's sole purpose is to output the Q-value, which appraises any given action in a way that encapsulates current and future rewards. In other words the actions are not only judged on its direct consequences but also the long term effects of said actions. The DDPG built upon this solution by also introducing a second network called the Actor, that takes state as an input and returns the optimal action. Additionally, DDPG adds a mechanism for exploration by adding noise to the output of the Actor during training. This noise is sampled from either a Gaussian distribution or an Ornstein-Uhlenbeck process; in this work Gaussian noise was used.

Both the Critic and Actor use an active network as well as a corresponding target network. Both active and target networks are initialized from their active counterpart, but are then updated separately. The active networks, that do the actual work, are updated through back-propagation while the target networks are updated through a so called "soft update". The soft update is defined as:

$$\theta_{\text{target}} \leftarrow \tau\, \theta_{\text{active}} + (1 - \tau)\, \theta_{\text{target}} \tag{4.1}$$

where θ_{target} is the target network, θ_{active} is its active counterpart and $\tau \in (0, 1)$ is a soft update rate.

The purpose of the target network is essentially to keep a version of the network that is updated slowly for stability, in order to allow the active network to update in a fast and volatile manner for exploration. This is relevant for how the active networks are updated, which unlike the target networks is different for the Actor and Critic. The Critic is trained to predict the current reward as well as future rewards based on an action, it does so by first calculating a target Q-value:

$$y_t = r_t + \gamma Q_{\theta_i'}\Big(s_{t+1},\ \mu_{\phi'}(s_{t+1})\Big) \tag{4.2}$$

where y_t is the target Q-value, r_t is the reward, γ is a discount factor, $Q_{\theta'}$ is the target Critic, s_{t+1} is the following state and $\mu_{\phi'}$ is the target Actor.

The loss function for the critic is defined as a Mean Squared Error (MSE) between the target Q-value and the output from the Critic:

$$\mathcal{L}_Q = \frac{1}{N} \sum_{i=0}^{N} (y_t - Q(s_t))^2 \tag{4.3}$$

where N is the batch size.

The Actor is trained to maximize the Q-value, therefore its loss function is defined as the negative of the output of the Critic

$$\mathcal{L}_\mu = -Q(s_t) \tag{4.4}$$

Because the Actor's loss is defined in terms of the critic's output rather than the actor's direct predictions, the actor parameters must be updated via the policy-gradient. Specifically,

$$\nabla_\phi J \approx \mathbb{E}_{s\sim\mathcal{D}}\Big[\nabla_a Q_\theta(s,a)\Big|_{a=\mu_\phi(s)} \nabla_\phi \mu_\phi(s)\Big] = \frac{\partial Q_\theta}{\partial a}\Big|_{a=\mu_\phi(s)} \frac{\partial \mu_\phi(s)}{\partial \phi} \tag{4.5}$$

In order to train both networks through back-propagation in batches, experiences are saved in tuples of $(s_t, a_t, s_{t+1}, r_t, d_t)$. The experience tuples are stored in a replay buffer. During training, experiences are sampled randomly following a uniform distribution for back-propagation. An important mechanism for the replay buffer is a warm up period in which the buffer is filled but the agent is not trained. This ensures that the same initial experiences are not trained on repeatedly, avoiding early training bias loop. Hyper parameters for the replay buffer as well the rest of the algorithm are displayed in table 4.1. DDPG laid the groundwork for RL algorithms with continuous state and action space, however its over-optimistic critics and volatile updates motivated the enhancements in TD3.

4.2.2 TD3

The TD3 algorithm was introduced to improve upon some of the shortcomings of DDPG, one of which is the inherent optimism of the Critic network θ. To combat this, TD3 introduced the Twin-Critic system in which the agent keeps two critics from which the Q-value is inferred, only the lower value out of the two is then used in order to keep a level of pessimism. Secondly, to add more stability to the training the TD3 introduced adding noise for updating the networks. This noise is used as part of a Target Policy Smoothing in which noise is added to the target policy during training, in order to avoid overfitting of the Critic. Finally, TD3 also added a delay in the update of the Actor network so that the Actor is only updated every other or third update step. This is done to control volatility in the actions by letting the Critic explore slightly more before using it to update the Actor. Algorithm 1 shows the workflow of the TD3 implemented in this work while table 4.1 show the hyper-parameters used.

Algorithm 1 Twin-Delayed Deep Deterministic Policy Gradient (TD3)

Initial actor parameters ϕ, critic parameters θ_1, θ_2, empty replay buffer $\mathcal{D}$
Initialize target networks: $\mu' \leftarrow \mu$, $\theta'_1 \leftarrow \theta_1$, $\theta'_2 \leftarrow \theta_2$
for each interaction step **do**
  Observe state s_t, sample action with exploration noise:

$$a_t = \mu(s_t) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma)$$

  Execute a_t, observe the next state s_{t+1}, reward r_t and done signal d. Store $(s_t, a_t, r_t, s_{t+1}, d_t)$ in $\mathcal{D}$
  if time to update **then**
    for $j = 1 \ldots N_{\text{updates}}$ **do**
      Sample minibatch $\{(s, a, r, s', d)\} \sim \mathcal{D}$
      Compute noisy target action:

$$\tilde{a}' = \text{clip}\Big(\mu_{\phi'}(s'),\, \epsilon',\, a_{\min}, a_{\max}\Big),\ \epsilon' \sim \text{clip}\Big(\mathcal{N}(0, \sigma), -c, c\Big)$$

      Compute target Q-value:

$$y = r + \gamma \min_{i=1,2} Q_{\theta'_i}(s', \tilde{a}')$$

      Update critics by gradient descent on

$$\mathcal{L}_{\theta_i} = \tfrac{1}{N} \sum (Q_{\theta_i}(s, a) - y)^2, \quad i = 1, 2$$

      if $j \,\%$ policy_delay $= 0$ **then**
        Update actor by policy gradient:

$$\mathcal{L}_\phi = -Q_1(s_t)$$

        Soft-update targets:

$$\phi' \leftarrow \tau\, \phi \ + \ (1 - \tau)\, \phi',$$
$$\theta'_i \leftarrow \tau\, \theta_i \ + \ (1 - \tau)\, \theta'_i.$$

      end if
    end for
  end if
end for

Table 4.1: TD3 Hyperparameters

Hyperparameter	Symbol	Value
Critic learning rate	α_c	10^{-3}
Actor learning rate	α_a	10^{-4}
Discount factor	γ	0.99
Soft-update rate	τ	0.005
Policy noise std.	σ	0.2
Noise clip range	c	$[-0.5, 0.5]$
Policy update delay	d	2
Replay buffer size	$\lvert\mathcal{D}\rvert$	10^6
Batch size	B	256

4.3 Problem Formulation: States, Actions, and Rewards

4.3.1 States and Actions

The central role of the environment is to emulate $P(s_{t+1} \mid s_t, a_t)$ which outputs the next state given an action, as well as a reward for that action as described in Section 4.1. The actions are control variables for the blower, heater and evaporator in the system described in 2.1.2. These components are regulated through their power consumption, therefore the action vector is defined as

$$a = \left[P_{\text{blower}},\ P_{\text{evap}},\ P_{\text{heater}}\right] \in [0, 1]^3 \tag{4.6}$$

The state needs to hold the information that the actor needs to make an informed decision, which for this aim is information regarding the cabin temperature. For this, dT, which is the difference between the average cabin temperature and the set-point temperature, will be used. The benefit of this definition is that the set-point can be changed at any point during the drive cycle and the actor will be able to adapt. In addition, the agent needs information on the available energy in relation to the projected energy consumption. The state vector will therefore also contain the current SoC, as well as the remaining distance in the drive cycle. Finally, since the agent is drawing power from a shared energy budget, information regarding how much energy is being drawn by the rest of the system will also be included. The reason for including the remaining distance instead of the projected energy consumption is that a reliable estimation of energy consumption for a drive cycle is never guaranteed, and so the hope is that the agent will learn a policy that safely estimates the energy budget for a given drive cycle with reliable information. Since the state and the action vectors will be used as inputs to a neural network, they will both be normalized.

4.3.2 Rewards

The main benefit of using RL over conventional control strategies is the inherent multi-objective-oriented controls RL provides. The mechanism through which the RL becomes multi-objective is the reward-shaping. As mentioned earlier, the goal of a MDP is to find a policy such that the cumulative rewards are maximized. Subsequently, any objective embedded into the rewards will be maximized. In the case of multiple

conflicting objectives, the algorithm finds a solution with optimal balance. Convergence on an optimum policy is therefore not possible without a balanced reward function. In broad terms, the total reward is defined as the weighted sum of the rewards for each objective:

$$r = \lambda_T r_{\text{comfort}} + \lambda_P r_{\text{energy}} \tag{4.7}$$

where r_{comfort} and r_{energy} are the rewards for thermal comfort and energy efficiency respectively while λ_T and λ_P are their respective scalar weights.

In order to simplify the definition of r_{comfort} and r_{energy} and ensure optimal behavior, each objective is split into sub-objectives. The sub-objectives are mainly based on an intuitive understanding of the requirements of the agent as well as domain knowledge and empirical reasoning. For energy efficiency, the sub-objectives are two; minimize energy consumption and enforce smooth actions. These are defined as:

$$r_{consumption} = \frac{a_t^{(2)}}{COP} + a_t^{(3)} \tag{4.8}$$

$$r_{smooth} = \frac{1}{3}\sum_{i=1}^{3}\left(a_t^{(i)} - a_{t-1}^{(i)}\right)^2 \tag{4.9}$$

where $a_t^{(2)}$ is the evaporator load, COP is the Coefficient of Performance (COP) for the compressor, $a_t^{(3)}$ is the heater load. The energy consumption from the blower is not included in the reward as it is negligible compared to the evaporator and heater load.

For thermal comfort, the sub-objectives are two as well; the difference between the cabin temperature and the adaptive setpoint. The adaptive setpoint is defined as the true setpoint plus a offset that depends on the energy budget, ρ.

$$\rho = \frac{E_{available}}{E_{needed}} \tag{4.10}$$

$$\delta(\rho) = \delta_0 \cdot \frac{1}{1 + e^{\left(k(\rho - \rho_{\text{ref}})\right)}} \tag{4.11}$$

$$T^* = T_{setpoint} + \delta(\rho) \tag{4.12}$$

where E_{needed} is the projected energy consumption, $E_{available}$ is the usable energy budget in the batteries, δ_0 is the upper bound of the offset and ρ_{ref} and k are parameters to smoothen the function.

The projected energy consumption E_{needed} is calculated based on the average total energy consumption for the vehicle based on distance, calculated from real data. The temperature reward is then calculated based on dT which was defined earlier as the difference between the cabin temperature and the adaptive setpoint.

$$r_{temperature} = -dT \tag{4.13}$$

The offset function $\delta(\rho)$ is defined such that the offset grows once the energy slack ρ falls below the reference value. To aid with exploration, the second thermal sub-objective is a bonus when the temperature is close to the adaptive setpoint. The final reward is therefore defined as

$$r = \lambda_T(r_{\text{temperature}} + r_{\text{bonus}}) + \lambda_P(r_{\text{consumption}} + r_{\text{smooth}}) \tag{4.14}$$

Individual weights, λ_T and λ_P, are empirically determined. Note that this heuristic reward formulation serves as a starting point to test adaptability; determining the optimal offset bounds and reward weights lies beyond the scope of this thesis and is proposed as future work. The parameters used in the reward function are for ensuring functionality.

4.4 Environment Metamodel

The straightforward way to infer the temperature from the environment given an action is to model the system and run simulations for each interaction. An approach is using a model of the system in a simulation software such as GT suite. However, the issue with this approach is that RL inherently is very time consuming as the agent needs to interact with the environment a substantial amount of times in order to converge on a policy. Since inference from complex models are also time consuming, especially when delegated to a separate simulation software, it becomes a major bottleneck in the process. For that reason, a fast-feedback environment is vital for feasibility [5].

Therefore, the approach used in this thesis is creating a metamodel for this task by training a neural network to predict the environments transition dynamics $f_\Omega(s_t, a_t)$ so that

$$\hat{s}_{t+1} = f_\Omega(s_t, a_t) \tag{4.15}$$

where $\hat{s}_{t+1}$ is the predicted new state and f_Ω is an approximation of $P(s_{t+1} \mid s_t, a_t)$ This approach can offer the fast inference time necessary, however the challenge now lies in accuracy and reliability of f_Ω. This section will focus on the methodology pertaining the metamodel used to predict the thermal response of the environment. This includes descriptions for the dataset generation and the architecture for the model itself.

4.4.1 Data Generation

The purpose of the neural network is to mimic the simulation model and so a synthetic dataset created through simulation is sufficient. The algorithm for creating the dataset as well as the statistical properties of the data, are vital for ensuring proper metamodel performance. The simulation model used in this study is a preexisting, validated 1D model in GTsuite created by Volvo Groups Thermal Department.

The data was generated by creating pseudo-random waveforms for each input which was fed to the model. The GT suite model simulates the system and documents the necessary input and output signals. The model expects six inputs features: Electrical current to the blower, Evaporator Load, Heater Load, Bypass Valve position, Solar Load and Vehicle speed.

The waveforms are a combination of different forms the signals are expected to take during the RL training. These forms are determined based on experience and are as follows:

- Rapid jumping: Here the signal takes on a "bang-bang" behavior with sudden jumps between different values.
- Oscillations: The signal oscillates with a random amplitude and phase.
- Ramp: A slow linear rise or fall between two random values.
- Plateau: The signal takes a single value for the whole section.

For all of waveforms, noise is added to the signals. A number of different waveforms is created for each of the input variables which are then combined randomly using latin hypercube sampling. Each sample is then fed to the model for which it simulates the system while logging the output features, which are the average cabin air temperature as well as the intermediate temperatures seen in figure 2.1. These are the temperatures before and after both the heater and evaporator as well as the temperature in the mixing area directly before entering the cabin. This is done in different ambient conditions with the aim that the metamodel will also be able to interpolate to unseen ambient conditions. A variable that will not be considered in detail is the soak temperature, which is the initial temperature of the cabin. For all the simulations the soak temperature will be kept equal to the ambient temperature for simplicity.

By making sure the initial waveforms span the entire input range, the resulting dataset remains broadly representative of the full expected ranges for each signal. Additionally, the overall frequency of the states represented in the dataset is controlled to be similar to a Gaussian distribution with the expected value equal to the average in real-world scenarios.

4.4.2 Twin Fourier Neural Operator

This thesis introduce a twin-model architecture that integrates a Physics-Informed Neural Network (PINN) for auxiliary variable prediction with a Fourier Neural Operator (FNO) for the primary output. This architecture specialized in physics based timeseries predictions in which the laws of physics are partially enforceable. Inference involves predicting auxiliary outputs and using these features as inputs for predicting the main output, the average cabin air temperature. The auxiliary outputs in this work is the intermediate temperatures mentioned above. This involves creating a model with a shared spine and separate heads with a flow of information from the first head into the next as shown in figure 4.2. A FNO network was chosen as a base, due to their superior capabilities of capturing partial differential equations [7]. The key to this approach is that predictions of the main output has through previous experience shown to be extremely accurate when using the auxiliary data as inputs, but these features will not be available at inference time. At the same time, enforcing conservation loss over single components is simple using the load for the given component, however the physics in the cabin are more complicated. As mentioned in section 2.1, in order to define a heat equation for the cabin all thermal loads need to be considered. Comparatively for a single component only the load of that component affect the flow. Using a PINN, a high accuracy of the auxiliary head can be achieved, which will assuredly lead to high overall accuracy as the first head is the limiting factor. A PINN version of the PINN has already been studied and proven to improve reliability [6]. The loss function is derived from the energy conservation law as shown in section 4.4.3.

The framework for FNO first introduced by Li et al. [7], was modified for a single spacial dimensional in order to fit the nature of the task at hand. Essentially, the task is a multivariate one-dimensional time-series prediction for forecasting average cabin temperature using auxiliary data as both outputs and inputs. As a result of the temporal relationship in the data, a sliding-window approach was used in order to provide enough historical context for each prediction. The historical context is controlled by the window size parameter. Due to thermal inertia, a large amount of historical context is necessary to capture the thermodynamics of the system. However, accuracy is also expected to decrease with a prediction horizon and for this reason the windows cannot be allowed to be non-overlapping. Using overlapping windows allows for the possibility of picking a prediction horizon separate from the window size, thereby conserving accuracy while allowing the model enough historical context to capture the thermal inertia of the system. Since the window size is closely related to the thermal inertia it can be determined by looking at the response time of the system. The stride parameter, which is equivalent to the prediction horizon, is determined based on the predictive capability of the model given a certain window size.

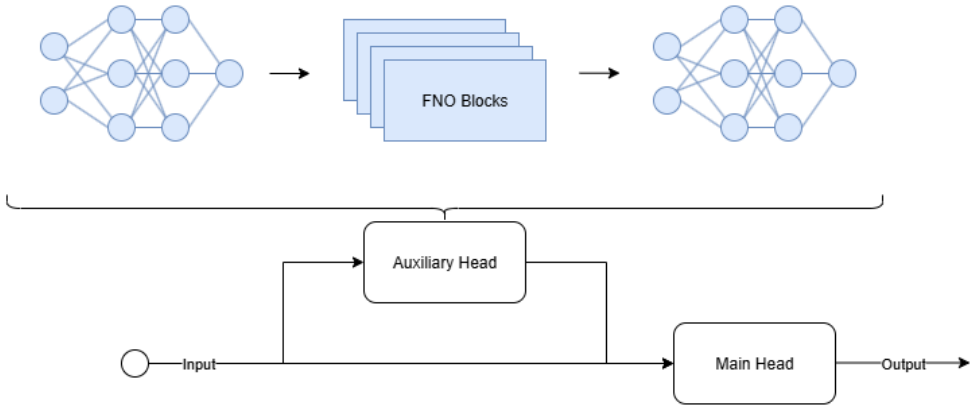


Figure 4.2: Twin FNO Architecture. The inputs flow into the auxiliary head which predicts the auxiliary outputs that are then appended to the original input before inference through the main head, that predicts the final output. Each head is a FNO model in itself, consisting of both linear layers and FNO blocks.

4.4.3 Physics Loss Function

A PINN is essentially a neural network which uses a second loss function which is based on the laws of physics. This loss function enforces the correct behavior and thereby enhances learning without additional data. The physics loss function is used in addition to the traditional data loss to calculate the overall loss. In this case, the physics loss function consists of three equations that was applied to the outputs of the auxiliary head. The four outputs are as follows: $T_{e,i}$, $T_{e,o}$, $T_{H,o}$ and T_{mixing}. Note that the temperature before the heater $T_{H,i}$ is equivalent to the temperature out of the evaporator $T_{e,o}$. As mentioned above, the loss function is derived from the conservation law:

$$\dot{Q} = \dot{m} c_p \Delta T \tag{4.16}$$

where $\dot{Q}$ is the energy rate added to the fluid, $\dot{m}$ is the mass flow rate, c_p is the heat capacity of the fluid and ΔT is the difference in temperature across the component.

Applying this equation to both the evaporator and the heater separately gives us equations that can be used to calculate the temperature out of the components.

$$T_{e,o} = T_{e,i} + \frac{\dot{Q}_{evaporator}}{\dot{m}_{total} c_p} \tag{4.17}$$

$$T_{H,o} = T_{e,o} + \frac{\dot{Q}_{heater}}{\dot{m}_{heater} c_p} \tag{4.18}$$

where m_{total} is the total mass flow rate, calculated from the electrical current to the blower and m_{heater} is the fraction of mass flow rate that flows through the heater. This is calculated with the position of the bypass valve.

Finally, T_{mixing} is given by the law of the lever:

$$T_{mixing} = \frac{(\dot{m}_{total} - \dot{m}_{heater}) T_{e,o} + \dot{m}_{heater} T_{H,o}}{\dot{m}_{total}} \tag{4.19}$$

Applying these three equations, 4.17, 4.18 and 4.19, the physics of the system can be enforced to improve the fidelity of the auxiliary head.

Chapter 5

Results

This chapter covers the results found through the provided methodology. Section 5.1 validates the surrogate metamodel's accuracy and reliability while Section 5.2 highlights the results of the RL training and the final policy. Results on the final policy will focus entirely on the same ambient conditions for fair comparisons. Remaining ambient conditions all show similar results

5.1 Surrogate Model

The Twin FNO model was trained for 1000 epochs with an early stopping criterion that executes when the validation loss plateaus for 100 epochs. The model is evaluated on a held-out test set of unseen drive cycles. On this test data, the model predicted average cabin temperature with a mean absolute error of approximately $0.55°$C.

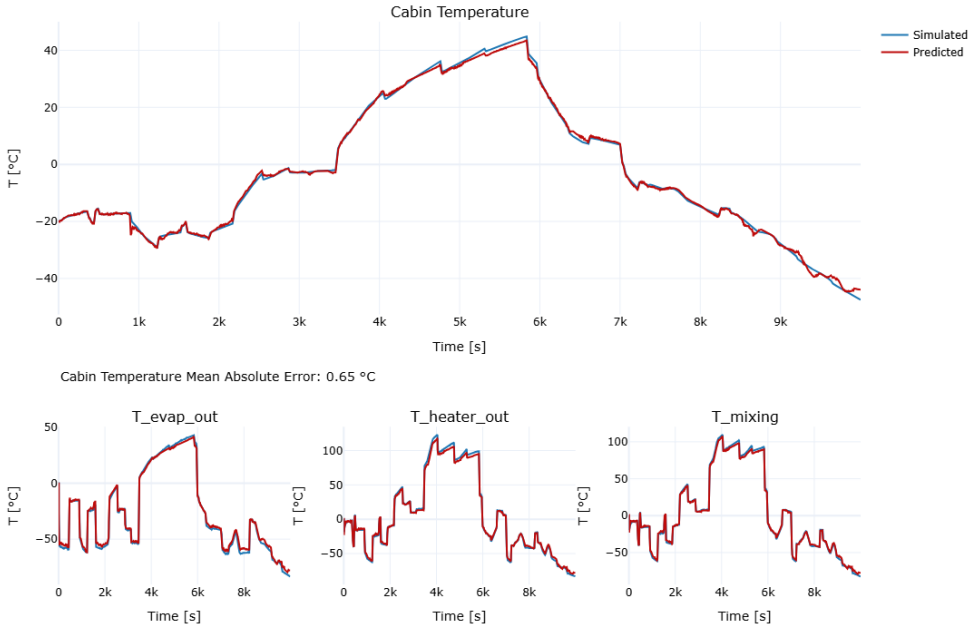


Figure 5.1: Twin FNO Surrogate metamodel accuracy on a test drive cycle. Ground-truth cabin temperature (Blue) versus Twin FNO prediction (Red). First row shows cabin temperature while the three plot on the second row show auxiliary predictions.

The dataset was created for only a few ambient conditions, with the hope that the model would be able to interpolate and possibly even extrapolate. Figure 5.2 shows the losses of the model across a spectrum of ambient temperatures. The graph shows the average loss to be substantially better for the specific conditions it has been trained on, $-20°C$, $0°C$, $20°C$, and $40°C$. For these conditions the average MAE is close to $0.55°C$ as per the results shown from the test set mentioned above. For the remaining conditions the average loss is higher, approximately $3°C$. The confidence interval, also seen in figure 5.2, shows similar behavior, very confident under specific ambient conditions while less so for remaining conditions. Because the surrogate's prediction error rises sharply outside the ambient temperatures on which it was trained, the RL experiments reported in section 5.2.1 are restricted to these four ambient conditions to ensure reliable feedback during policy learning.

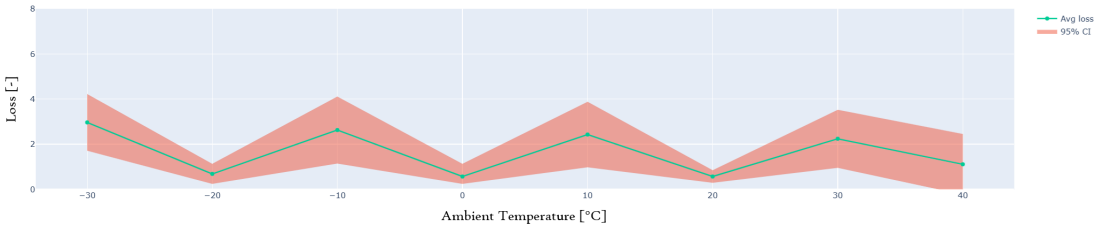


Figure 5.2: Performance of Twin FNO model across varying ambient conditions. Plot shows the performance is considerably better for certain ambient temperatures, $-20°C$, $20°C$, $0°C$, and $40°C$ while still showing reasonable performance for the rest.

Figure 5.3 shows how the model performs across varying soak temperature given a constant ambient temperature. For $T_{soak} \neq T_{amb}$ the average loss explodes while showing good performance for $T_{soak} = T_{amb}$. The graph shows similar but more severe behavior compared to figure 5.2. Outside of the scope that the model was trained on, it is not functional. Note that this figure is included solely to illustrate why the soak temperature is held equal to the ambient in all subsequent experiments; the large errors seen for $T_{soak} \neq T_{amb}$ would otherwise add substantial complexity to the learning problem.

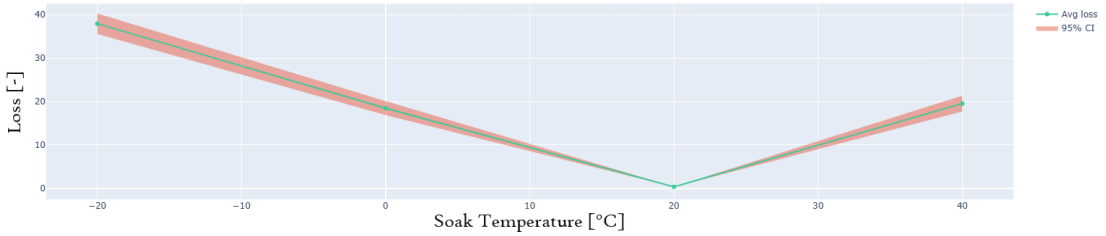


Figure 5.3: Performance of Twin FNO model across varying soak temperatures given an ambient temperature of 20°C. Plot shows the loss diverges as it moves away from the operating point, $T_{soak} = 20°\text{C}$.

5.2 Agent

Figure 5.4 visualizes the agent's ability to learn a policy. The early decline reflects initial exploration, while the sharp rise corresponds to the agent discovering effective temperature control heuristics. The early plateau suggests limited further gains, most likely due to only small changes being made to the policy.

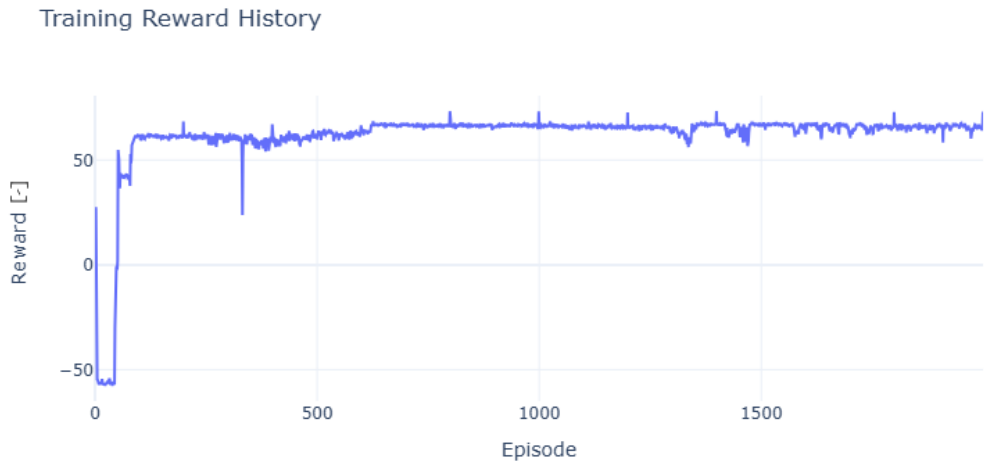


Figure 5.4: TD3 learning curve for the cabin climate agent. Episode return versus environment steps: an initial exploration dip is followed by a sharp spike and plateau, indicating convergence to a stable policy after 600 episodes.

5.2.1 Policy

The figures below are a further look into the final policy on which the agent converges. Figure 5.5 displays a warm-up cycle in which the initial charge is close to the full

capacity of the battery and the energy budget is more than sufficient ($\alpha >> 1$). The drive cycle is initialized with a SoC of 90% while the cabin temperature is $22°$C below the setpoint temperature. In this case, the agent controls the thermal regulators in such a way that the cabin temperature reaches the setpoint in the least amount of time while still minimizing energy consumption. The evaporator is turned off for the full drive cycle while the heater is controlled in such a way as to reach the setpoint as fast as possible and from there stick to the setpoint. In the initial stages where the temperature error is the largest, the heater is fully activated, after which the load is decreased to very small oscillations to combat the natural drop in temperature throughout the drive cycle.

Figure 5.6 highlights how the policy changes under tougher conditions with respect to the energy budget. In this drive cycle, the conditions are kept identical to the previous example, except that the initial SoC is 20%. The policy is similar to the first case with regard to the profile of the actions. The important distinction is the policy in the early stages of the drive cycle. In this example, the heater load drops sharper and earlier. The result is that the cabin temperature equilibrium is slightly lower, converging on a temperature that deviates from the setpoint. This leads to a lower energy consumption by compromising on thermal comfort in a case of low SoC. In total, the low SoC policy consumes 30% less energy with a $5°$C deviation.

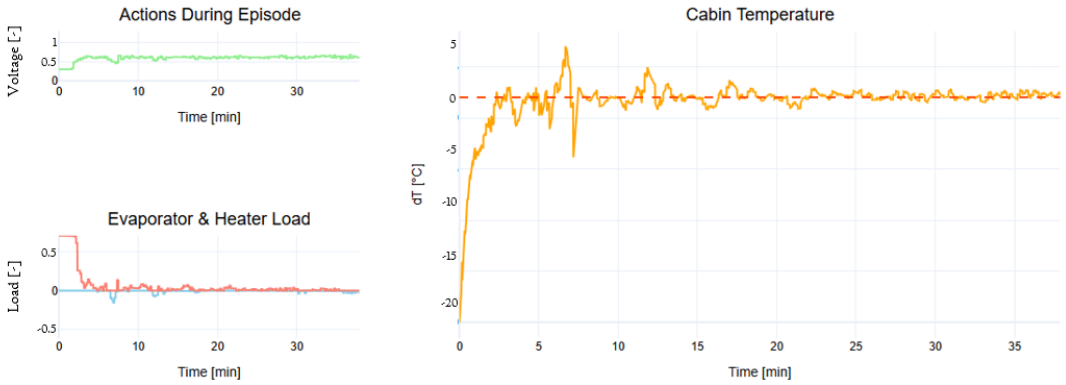


Figure 5.5: Comfort focused policy. Cabin temperature (orange), evaporator (blue), blower (green) heater (red) effect over time. The agent fully utilizes the heater early, then tapers power to hold the set-point, while keeping the evaporator off; the blower profile resembles the heater with the exception that it stays on the whole drive cycle.

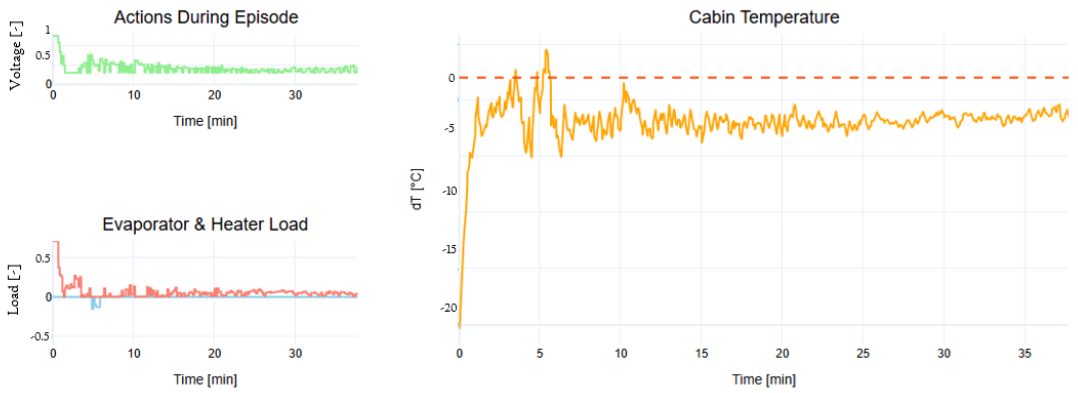


Figure 5.6: Energy-saving policy. To conserve charge the agent reduces heater load, allowing the cabin temperature to stabilize 5 °C below the set-point and cutting HVAC energy by 30% versus Fig. 5.5.

In order to understand the full range of the agents adaptability, figure 5.7 visualizes the full range of temperature offsets possible for drive cycles in the same conditions with respect to expected distance and ambient temperature. All the simulations used to generate this graph were for warm-up cycles. The graph shows the maximum temperature offset is 6°C which results in a 40% reduction in HVAC load at equilibrium. Crucially, the drive cycles in each data point for this plot have similar blower policies which is independent from the SoC.

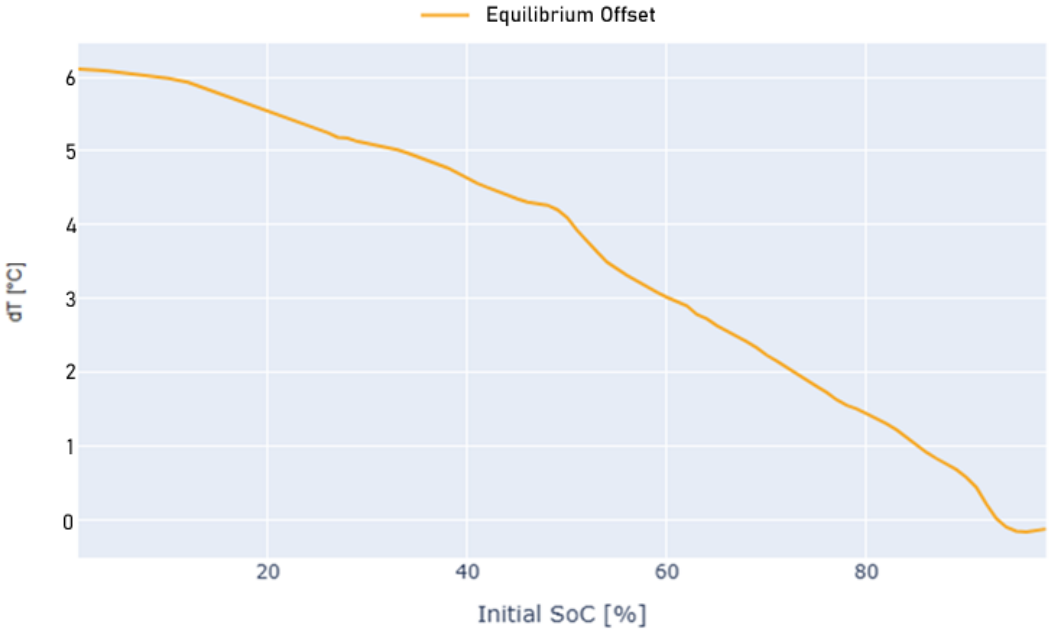


Figure 5.7: Equilibrium policy. The graph shows how the offset between the equilibrium temperature and the setpoint temperature changes as a function of energy availability.

Chapter 6

Discussion

6.1 Twin FNO Surrogate Model

An important aspect of RL is that the agent is initialized randomly, meaning that the first few actions are random. This is further boosted by the initial training being dominated by exploration followed by a long period of exploitation, as seen in figure 5.4. Very importantly, this behavior is typical within RL, reinforcing the notion that the agent will only converge on a non-trivial policy if it finds success in the early exploration stage. The inherent randomness and the importance of the early stages further emphasizes why it is so vital that the metamodel is able to accurately handle any and all types of temperature profiles. If the metamodel fails in the early stage, then it is impossible to properly reward favorable patterns that can function as the foundation of the final policy. Conversely, it is impossible to punish inefficient patterns so as to discourage the agent from exploiting it. The surrogate metamodel's Mean Absolute Error (MAE) of $0.55°\text{C}$ indicates that the Twin FNO architecture captures the cabin's thermal dynamics with an exceptional level of accuracy. Given the pseudo randomness of the data, as can be seen on the temperature profiles in figure 5.1 and in appendix A, it is expected that the accuracy will carry over to the randomness in the early stages of RL. This is further backed by the high confidence seen in figure 5.2 at the specific ambient temperatures $-20°\text{C}$, $0°\text{C}$, $20°\text{C}$, and $40°\text{C}$ which tells us that the model is reliable in these conditions.

Furthermore, figure 5.2 highlights aspects of the Twin FNO model that are important to consider. While the behavior in the trained ambient conditions are exceptional, the assumption that this would carry over to other ambient conditions has been partially incorrect. The graph shows a significant increase in average loss and decrease in confidence in all ambient conditions except for the ones in which the model has been trained. But, while the increase is significant the model still holds a MAE of approximately $3°\text{C}$. This means the model is still capturing the dynamics of the cabin. More notably figure 5.3 shows that this is not the case for $T_{soak} \neq T_{amb}$. The loss here explodes in a way that shows the model is completely failing outside trained conditions. The working theory is that since the dataset contains a few different ambient conditions, the change in T_{amb} is not a foreign concept to the model, but it cannot capture the exact effect without specific training. On the other hand, the whole dataset never distinguishes between T_{soak} and T_{amb}. Figure 5.2 and 5.3 provides a more nuanced picture of what

the Twin FNO model is capable of. Even with these shortcomings however, the model performs exceptionally for the intended use and it is believed the twin headed approach is the reason for this.

6.2 Policy Trade-offs

Figure 5.4 highlights important patterns that explain the current performance of the agent; however, it also shows shortcomings that can be improved upon. The agent's learning curve, with its initial exploration dip followed by a sharp performance rise, reflects an effective balance between exploration and exploitation under reward weights. Yet the early plateau hints at a potential ceiling imposed by the current reward-shaping formulation. Tuning or restructuring the reward function could potentially unlock further improvements. By looking at individual drive cycles seen in figures 5.5 and 5.6, it is clear that there is still room for improvements. The agent has trouble learning the finer details as it still oscillates instead of converging to a load that is equal to the sum of internal and external loads for a true equilibrium. The oscillations are most likely due to either limited incentives in the reward function or a limitation in the hyper parameters, such as learning rates.

Policy behavior under changing energy availability highlights the controller's ability to trade off comfort for energy savings when necessary. The 30% reduction in energy consumption, between figure 5.5 and 5.6, at the cost of a $5°$C comfort deviation under low-SoC conditions demonstrates responsiveness, but also raises questions about acceptable comfort violations in real-world deployments. The reward shaping is based on setting a value to each degree of deviation from the setpoint based on the energy availability, however the profile of that value function is an important point of discussion. Figure 5.7 illustrates the value function, showing how the learned controller adapts its steady-state behavior to energy availability. As the SoC rises, the equilibrium cabin temperature falls almost linearly from about 6 °C from the set-point down to $0°$C. This confirms that the reward shaping achieves the intended trade-off: when energy is scarce the agent relaxes comfort, tolerating a cooler cabin to save power. But as soon as additional energy is available, it progressively allocates more heating to converge on the set-point. The monotonic trend demonstrates that the controller has learned a continuous, rather than threshold-based, mapping between SoC and comfort. This is something conventional rule-based logic would require manual tuning to replicate. However, whether this almost linear mapping is the optimal solution or not is another topic to investigate. This work has focused on using the TD3 framework for learning any acceptable form of adaptation, therefore I leave it for future work to find an improved methodology in which both the bounds of the offset function $\delta(\rho)$, as well as the profile are optimized. This would mean finding optimal values for δ_0, ρ_{ref} and k, or potentially finding a whole new function. Additionally, this work has limited the notion of thermal comfort to average temperature. Expanding this idea to other aspects such as humidity is an interesting problem to work through. Forming and answering question equivalent to how much 1 degree of deviation is worth in term of $[kW]$ when considering humidity is also a task for future work.

While a 40% reduction in energy consumption is substantial, it is important to put it

into perspective. The vast majority of energy consumption in BEV is due to propulsion. As stated earlier, energy consumption for thermal regulation varies between 14% and 18% depending on the source. This means that, in terms of total energy consumption, a 40% reduction in HVAC load is less significant, especially since this reduction occurs only under specific conditions. Consequently, it is unlikely that this work will lead to noticeable improvements on a charge-to-charge basis. Nevertheless, the cumulative reduction in energy consumption over the vehicle's lifetime could still reach meaningful levels. While this may not directly enhance vehicle performance or driver experience, it can contribute to lowering the overall environmental impact. Although this was not the primary objective of the work, it aligns with the broader goal of reducing the automotive industry's environmental footprint. Furthermore, while this work fills the previously identified gap in research regarding controlled HVAC for vehicles, it does explain why most of the research is focused on building HVAC. In buildings, HVAC is most likely the main energy consumer, which means 40% decrease in HVAC load is much more significant.

Chapter 7

Conclusion

Building on the aim of training an adaptive agent that jointly optimize passenger comfort and energy use, this thesis demonstrates that a TD3 algorithm, coupled to a Twin FNO surrogate, can discover non-trivial actuator sequences that reduce HVAC energy demand by about 40% by tolerating a deviation from the set-point when energy availability is low. The surrogate model, a twin-FNO, is an architecture specifically developed for predicting the thermodynamics of a HVAC system. It predicts auxiliary variables (the intermediate temperatures throughout the system) and uses these features to estimate the average cabin air temperature. This model captures the cabin thermodynamics with a mean absolute error of roughly 0.55°C and is therefore suitable as the virtual environment for training a TD3-based agent. The results show that the framework can train a controller capable of both precise temperature regulation and deliberate deviations to save energy, illustrating the feasibility of energy-adaptive climate control.

However, it is important to note that reducing HVAC load by 40% translates to only a single-digit percentage reduction in total vehicle energy use, because cabin climate control represents roughly $14 - 18\%$ of a battery-electric truck's energy consumption. Nevertheless, the cumulative reduction in energy consumption over the vehicle's lifetime could still reach meaningful levels. While this may not directly enhance vehicle performance or driver experience, it can contribute to lowering the overall environmental impact.

At the same time, the work has important limitations that temper its practical impact. The agent's performance is evaluated only under the soak conditions used to train the surrogate, since the surrogate itself fails when soak and ambient temperatures differ. Another major limitation is the narrow definition of thermal comfort, which here is reduced solely to average cabin temperature, omitting humidity and other relevant factors. Consequently, this implementation should be regarded as a proof of concept. Future work should broaden the scope to capture a fuller representation of thermal comfort and evaluate the method under more diverse conditions.

Chapter 8

Future Research

Building on the conclusions of this thesis, several directions for future work are identified below. I believe the main focus for future improvements should be perfecting the existing idea of adapting the weights of the reward function in such a way that the agent learns an inherent policy for adapting the action policy based on the available energy as well as other factors. Furthermore, investigating how adding more components from the HVAC system with the control of the agent could improve the results would be an interesting direction. For example the bypass valve and recirculation door has been kept constant in this work but these components are an important part of the system. These changes will most likely not only be resolved with alterations to the current reward function but also require changes in parameters as well as general setup of the TD3.

Appendix A

This appendix holds more results showcasing the performance of the Twin FNO surrogate model under different ambient conditions.

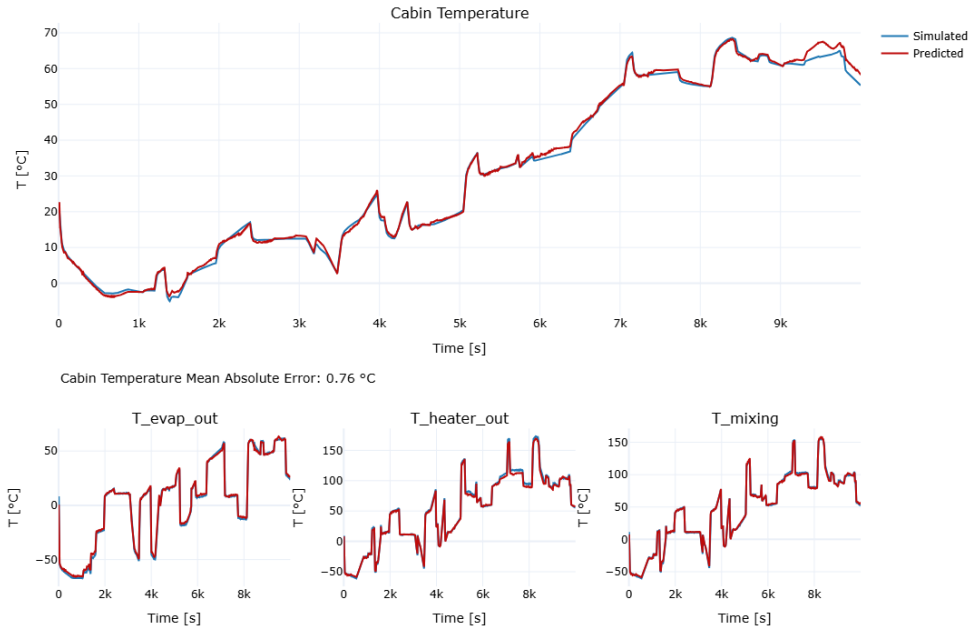


Figure A.1: Twin FNO Surrogate metamodel accuracy on a test drive cycle. Ground-truth cabin temperature (Blue) versus Twin FNO prediction (Red); rapid changes are tracked, demonstrating good fidelity under aggressive load swings.

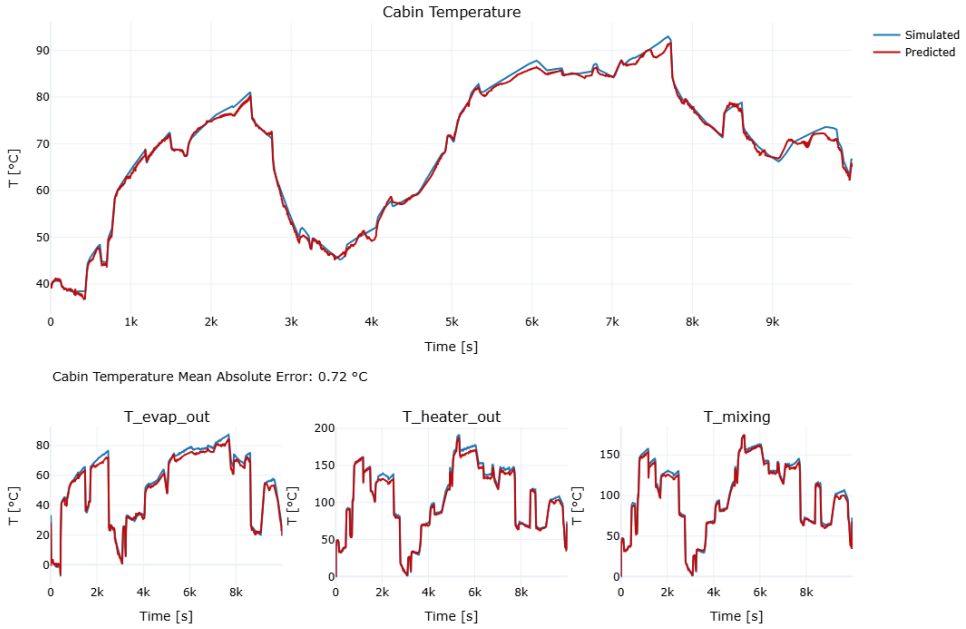


Figure A.2: Twin FNO Surrogate metamodel accuracy on a test drive cycle. Ground-truth cabin temperature (Blue) versus Twin FNO prediction (Red); rapid changes are tracked, demonstrating good fidelity under aggressive load swings.

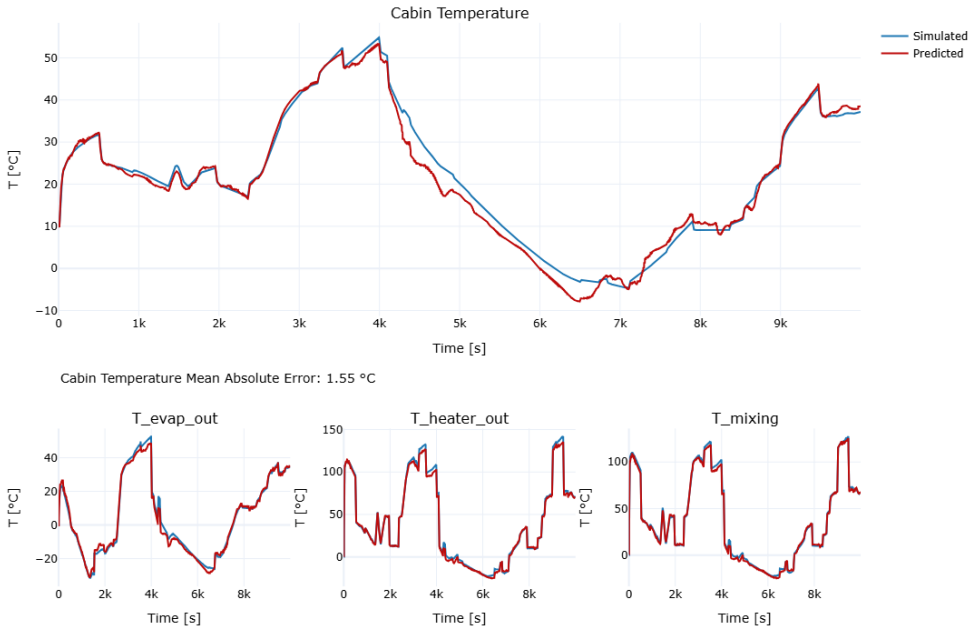


Figure A.3: Twin FNO Surrogate metamodel accuracy on a test drive cycle. Ground-truth cabin temperature (Blue) versus Twin FNO prediction (Red); rapid changes are tracked, demonstrating good fidelity under aggressive load swings.

Bibliography

[1] Marco Biemann et al. “Experimental evaluation of model-free reinforcement learning algorithms for continuous HVAC control”. In: *Applied Energy* 298 (2021), p. 117164. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2021.117164. URL: https://www.sciencedirect.com/science/article/pii/S0306261921005961.

[2] Sergi Esteban Bueno. “Vehicle HVAC system modeling and controlling”. UPC director: Domingo Biel Solé; Idneo directors: Jose Luis Blanco López, Marc Güell Brocal. MA thesis. Barcelona, Spain: Escola Tècnica Superior d’Enginyeria de Telecomunicació de Barcelona, Oct. 2021.

[3] Aisling Doyle and T. Muneer. “Energy consumption and modelling of the climate control system in the electric vehicle”. In: *Energy Exploration Exploitation* 37 (Jan. 2019), p. 014459871880645. DOI: 10.1177/0144598718806458.

[4] Scott Fujimoto, Herke van Hoof, and David Meger. “Addressing Function Approximation Error in Actor-Critic Methods”. In: *CoRR* abs/1802.09477 (2018). arXiv: 1802.09477. URL: http://arxiv.org/abs/1802.09477.

[5] Cheng Gao and Dan Wang. “Comparative study of model-based and model-free reinforcement learning control performance in HVAC systems”. In: *Journal of Building Engineering* 74 (May 2023), p. 106852. DOI: 10.1016/j.jobe.2023.106852.

[6] Gargya Gokhale, Bert Claessens, and Chris Develder. “Physics informed neural networks for control oriented thermal modeling of buildings”. In: *Applied Energy* 314 (May 2022), p. 118852. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2022.118852. URL: http://dx.doi.org/10.1016/j.apenergy.2022.118852.

[7] Nikola B. Kovachki et al. “Neural Operator: Learning Maps Between Function Spaces”. In: *CoRR* abs/2108.08481 (2021). arXiv: 2108.08481. URL: https://arxiv.org/abs/2108.08481.

[8] Jerry Luo et al. *Controlling Commercial Cooling Systems Using Reinforcement Learning*. 2022. arXiv: 2211.07357 [cs.LG]. URL: https://arxiv.org/abs/2211.07357.

[9] Antonio Manjavacas et al. “An experimental evaluation of deep reinforcement learning algorithms for HVAC control”. In: *Artificial Intelligence Review* 57.7 (June 2024), p. 173. DOI: 10.1007/s10462-024-10819-x. URL: https://doi.org/10.1007/s10462-024-10819-x.

[10] Tianshu Wei, Yanzhi Wang, and Qi Zhu. “Deep reinforcement learning for building HVAC control”. In: *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*. 2017, pp. 1–6. DOI: 10.1145/3061639.3062224.