Optimal Torque Control of Externally Excited Synchronous Motors by Reinforcement Learning

by

Nyi Nyi Aung

Submitted to the Department of Electrical Engineering, Electronics, Computers and Systems
in partial fulfillment of the requirements for the degree of
Erasmus Mundus Master Course in Sustainable Transportation and Electrical Power Systems
at the
UNIVERSIDAD DE OVIEDO
August 2024

Author . . . .
Nyi Nyi Aung

Certified by . . . .
Frank Schafmeister
Interim Professor
Thesis Supervisor

Certified by . . . .
Juan Rodríguez Méndez
Assistant Professor
Thesis Supervisor

Optimal Torque Control of Externally Excited Synchronous Motors by Reinforcement Learning

by

Nyi Nyi Aung

Submitted to the Department of Electrical Engineering, Electronics, Computers and Systems
on August 28, 2024, in partial fulfillment of the
requirements for the degree of
Erasmus Mundus Master Course in Sustainable Transportation and Electrical Power Systems

Abstract

Externally excited synchronous motors (EESMs) are a viable alternative to permanent magnet synchronous motors (PMSMs). They do not require rare-earth materials and offer an additional degree of freedom in the control structure through the rotor circuit.

Reinforcement learning (RL) offers several advantages over conventional controllers such as field-oriented control (FOC) or model predictive control (MPC). RL is model-free and data-driven, making it particularly useful for complex dynamic systems. Once adequately trained, RL can manage nonlinear behavior with, theoretically, optimal performance without the use of a complicated explicit model.

However, EESMs present a challenging control problem due to their complex dynamics and strong cross-coupling between axes. This makes it difficult for an RL agent to comprehend the drive's dynamic system and provide optimal actions within predefined constraints, such as current and voltage limitations. This thesis provides an initial proof of concept, demonstrating that a data-driven controller with proper reward design can effectively manage the intricate system of an EESM.

Thesis Supervisor: Frank Schafmeister
Title: Interim Professor

Thesis Supervisor: Juan Rodríguez Méndez
Title: Assistant Professor

Acknowledgments

This thesis is dedicated to my country, Myanmar, and its people, who continue to endure and show resilience during these challenging political times. I remain hopeful and committed to using the academic knowledge gained throughout this master's degree to actively contribute to my country's rebuilding and development once peace and stability are restored.

I would like to express my sincere gratitude to all those who have supported this work and contributed to my academic journey. First and foremost, I am deeply thankful to the coordinating professors of the "Sustainable Transportation and Electrical Power Systems (STEPS)" program for granting me the opportunity to join the "Erasmus Mundus Joint Master Degree" scholarship. Their belief in my potential has been invaluable.

My heartfelt thanks also go to the professors at Sapienza University of Rome, University of Nottingham, and University of Oviedo for their insightful lectures and guidance throughout the past two years. Their teachings have greatly enriched my academic experience.

I extend my special thanks to the LEA-AI team members at Paderborn University, especially Maximilian Schenke, Mario Peña, Darius Jakobeit, and Barnabas Haucke-Korber, for providing me with the opportunity to conduct my internship and master's thesis within their department. Their close guidance, along with the learning materials they provided, helped bridge my knowledge gaps in the field of reinforcement learning. This research would not have been possible without their invaluable support and supervision.

Last but not least, I would like to express my deep appreciation to my colleagues from the STEPS 2022-2024 cohort for sharing this two-year journey with me. Finally, I am profoundly grateful to my family, who have consistently stood by me, offering unwavering support in all circumstances.

Contents

1 Introduction **23**

1.1 Motivation . . . 23

1.1.1 EESM, an Alternative to PMSM . . . 23

1.1.2 Reinforcement-Learning-Based Control . . . 24

1.2 Objective and Structure of Research . . . 25

1.3 Conceptual Framework . . . 26

1.3.1 General EESM Modelling Approach . . . 26

1.3.2 Development of Electromagnetic Torque . . . 31

1.3.3 Current and Voltage Limit . . . 32

1.3.4 Optimal Operation of EESMs . . . 33

1.3.5 Power Electronic Converters Modelling . . . 33

1.3.6 Gym Electric Motor Toolbox . . . 36

1.3.7 Introduction to Reinforcement Learning and its Elements . . . 37

1.3.8 Introduction to Deep Deterministic Policy Gradient . . . 44

1.4 Basic Setup for training the agent . . . 50

2 Reinforcement Learning based Current Control **53**

2.1 RL-CCS Current Control . . . 53

2.2 Reward Design for CCS Current Control . . . 55

2.3 Training the DDPG Agent . . . 57

2.4 Validation and Performance Analysis . . . 63

2.5 Key Takeaways of the Chapter 65

3 Torque Control with Idealized Excitation Circuit Current Control 67

3.1 RL-CCS Torque Control 68

3.2 Controller Design Approach 69

3.2.1 Tuning the PI Controller 70

3.2.2 Reward Design for PI-assisted RL Control 71

3.3 Training the DDPG Agent 75

3.4 Validation and Performance Analysis 78

3.5 Key Takeaways of the Chapter 84

4 Torque Control with Reinforcement Learning based Excitation Circuit 87

4.1 Performance Priority Control 88

4.1.1 Reward Design for Performance Priority Control 89

4.1.2 Training the DDPG agent 91

4.1.3 Validation and Performance Analysis 96

4.2 Efficiency Priority Control 98

4.2.1 Insight to Efficiency 99

4.2.2 Reward Design for Efficiency Priority Control 100

4.2.3 Training the DDPG Agent 101

4.2.4 Validation and Performance Analysis 102

4.3 Comparative Analysis of Performance and Efficiency Priority Control 104

4.3.1 Performance Comparison 104

4.3.2 Efficiency Comparison 105

4.4 Key Takeaways of the Chapter 109

5 Conclusions and Outlook 111

5.1 Conclusions 111

5.2 Future Work 112

A Validation of RL Agent for CCS Current Control **115**

B Validation of PI-assisted RL Agent **117**

C Validation of Performance Priority Control Agent **121**

D Validation of Efficiency Priority Control Agent **123**

List of Symbols

i	Current
u	Voltage
Ψ	Flux
T, T_{em}	Electromagnetic torque
f_{s}	Sampling frequency
$\boldsymbol{T}$	Coordinate transformation matrix
k	Time step
ϵ_{el}	Electric rotor angle
ω_{el}	Electric rotor speed
ω_{mech}	Mechanical rotor speed
$R_{\mathrm{s}}, R_{\mathrm{f}}$	Stator and rotor resistance
$\boldsymbol{L}$	Inductance matrix
L	Absolute inductance
p	Machine pole pair number
U_{DC}	DC-bus voltage
σ_{l}	Leakage factor
τ	Time constant
$\mathcal{X}$	Set of states
$\mathcal{U}$	Set of actions
$\boldsymbol{\mathcal{P}}$	State transition probability
$\mathcal{R}$	Reward function
γ	Discount factor
g	Return
J	Expected return
π, π^*	Policy and optimal policy
q, q^*	Action-value function and optimal action-value function
$\boldsymbol{w}$, $\boldsymbol{w}^-$	Critic weights and delayed critic weights

$\boldsymbol{\theta}, \boldsymbol{\theta}^-$	Policy weights and delayed policy weights
$\boldsymbol{\mathcal{D}}$	Replay buffer
λ	Equivalent filter constant
α	Step size
μ	Mean of Gaussian noise sample
σ	Standard deviation of Gaussian noise sample
x	Scalar variable or parameter
$\tilde{x}$	Normalized variable or parameter
$\boldsymbol{x}$	Vector variable or parameter
$\boldsymbol{X}$	Matrix variable or parameter
$\dot{x}$	Time derivative of variable or parameter
x^*	Reference variable (except for π^* and q^*, see above)
$\hat{x}$	Estimated variable or parameter
$x_{\mathrm{sd}}, x_{\mathrm{sq}}$	Variable or parameter in the rotor synchronous d- and q-axes
$x_{\mathrm{a}}, x_{\mathrm{b}}, x_{\mathrm{c}}$	Variable or parameter in the stationary a-, b-, and c-axes
$\boldsymbol{x}_{\mathrm{abc}}, \boldsymbol{X}_{\mathrm{abc}}$	Variable or parameter in the stationary abc-reference frame
x_{α}, x_{β}	Variable or parameter in the stationary α- and β-axis
$\boldsymbol{x}_{\alpha\beta}, \boldsymbol{X}_{\alpha\beta}$	Variable or parameter in the stationary $\alpha\beta$-reference frame
x_{f}	Excitation circuit variable or parameter
$\boldsymbol{X}^{-1}$	Matrix inverse
$\boldsymbol{X}^{\dagger}$	Matrix pseudo-inverse
$\mathrm{diag}(\boldsymbol{x}_{1\times n})$	Matrix $n \times n$ with the terms of vector $\boldsymbol{x}$ in the diagonal and zeros elsewhere

List of Acronyms

4QC	4-quadrant converter
ANN	Artificial neural network
CCS	Continuous control set
DDPG	Deep deterministic policy gradient
EESM	Externally excited synchronous motor
FCS	Finite control set
FOC	Field-oriented control
GEM	Gym-electric-motor
MDP	Markov decision process
MPC	Model predictive control
MTPC	Maximum torque per current
MTPCL	Maximum torque per copper losses
PID	Proportional–integral-derivative (controller)
PI	Proportional-integral (controller)
PMSM	Permanent magnet synchronous motor
RL	Reinforcement learning
RMSE	Root mean square error

List of Figures

1-1 Schematic representation of EESM [1] 27

1-2 MTPCL current trajectories 34

1-3 Four quadrant converter 35

1-4 B6 bridge converter 35

1-5 Simplified structure diagram of an electric drive system 36

1-6 General disciplines of machine learning 39

1-7 The basic RL operation principle [2] 40

1-8 Visualization of neurons of an ANN [3] and exemplary activation functions . 46

1-9 Visual summary of DDPG working principle [3] 47

2-1 Schematic of overall CCS current control structure 54

2-2 Variation of the standard deviation of the Gaussian distribution over time steps 59

2-3 Reference inputs for an exemplary episode in the CCS current control structure 61

2-4 Average reward and episode length over the training steps 62

2-5 Reference inputs to a validation environment 63

2-6 Validation results of the agent at a speed of $\omega_{\text{mech}} = -20\text{s}^{-1}$ 64

3-1 Schematic of the overall CCS torque control structure with PI controller . . 68

3-2 Schematic of PI controller with plant 70

3-3 Schematic of the reward design approach 72

3-4 Reference inputs for an exemplary episode in the CCS torque control structure 77

3-5 Average reward and episode length over the training steps 78

3-6 Reference inputs to a validation environment 79

3-7 Validation results of the agent at a speed of $\omega_{\text{mech}} = -200\text{s}^{-1}$ 80

3-8 Validation results of the agent at a speed of $\omega_{\text{mech}} = -20\text{s}^{-1}$ 81

3-9 Validation results of the agent with i_{f}^{*} as a ramp input at a speed of -20s^{-1} 83

4-1 Schematic of the overall CCS torque control structure with RL-based excitation circuit 88

4-2 Schematic of the reward design approach for performance priority control . . 89

4-3 Variation of the standard deviation of the Gaussian distribution over time steps for dq- and f-actions 93

4-4 Reference input for an exemplary episode in the CCS torque control structure 95

4-5 Average reward and episode length over the training steps 95

4-6 Reference torque to a validation environment 96

4-7 Validation results of performance priority control at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$ 97

4-8 Reward corresponds to reference torque isoline region $\mathbb{A}$ 98

4-9 Variation of the standard deviation of the Gaussian distribution over time steps for dq- and f-actions 101

4-10 Average reward and episode length over the training steps 102

4-11 Validation results of efficiency priority control at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$ . . 103

4-12 Step reference torque input for efficiency analysis at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ 105

4-13 Efficiency analysis of performance priority control at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ 106

4-14 Efficiency analysis of efficiency priority control at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ 107

4-15 Optimization of different operating points at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ . . . 108

4-16 Efficiency comparison between two strategies for different operating points at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ 109

A-1 Validation results at a speed of 200s^{-1} for CCS current control 116

B-1 Validation results at a speed of $\omega_{\text{mech}} = 200\text{s}^{-1}$ for PI-assisted RL control . . 118

B-2 Validation results at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$ for PI-assisted RL control . . . 119

C-1 Validation results at a speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$ for performance priority control 121

D-1 Validation results at a speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$ for efficiency priority control 123

List of Tables

1.1 Comparison of PMSM and EESM [4] 24
1.2 Parameterization of the considered drive system, EESM [5] 38
1.3 General purposes of different branches of machine learning 39
1.4 Software components and versions 51
1.5 Detail data of workstation 51

2.1 Selected hyper-parameters set for CCS current control 58
2.2 Artificial neural network configuration for CCS current control 60

3.1 Parameters tuning PI 71
3.2 Selected hyper-parameters set for PI-assisted RL control 76

4.1 Selected hyper-parameters set for performance priority torque control 92
4.2 Artificial neural network configuration for CCS torque control 94

Chapter 1

Introduction

1.1 Motivation

The current research, titled “Optimal Torque Control of Externally Excited Synchronous Motors by Reinforcement Learning”, aims to control this AC drive using RL instead of conventional PID controllers or MPC. The following sections summarize the motivation for the choice of an EESM drive as the controlled system and the choice of RL as a control strategy.

1.1.1 EESM, an Alternative to PMSM

To the author’s best knowledge, PMSM is the most efficient electrical machine in the industry and is widely used in a variety of applications, including electric vehicles and industrial automation systems. Despite its power density and precise controllability, there are some downsides that should be taken into consideration. First and foremost, PMSMs are expensive compared to other AC drives because permanent magnets are used in the rotor. High-performance magnets are mostly made from rare earth elements, and the mining and processing of these materials raise environmental concerns. Moreover, only a limited number of rare-earth material suppliers are available, with China accounting for up to 70% of production in 2022 [6]. Therefore, if problems are encountered in Chinese production, it could lead to supply chain issues. Additionally, PMSMs are highly sensible to temperature; operating at high temperatures can lead to demagnetization, resulting in reliability issues. These challenges, along with constraints such as durability, and long-term reliability [6], have led to extensive research into alternatives to PMSMs, with EESMs being one of them. The comparison between EESM and PMSM is visualized in table 1.1.

Feature	**PMSM**	**EESM**
Rotor	Permanent magnets	Excitation winding
Excitation source	No external source	External DC supply
Efficiency	High (absence of rotor losses)	Slightly less than PMSM
Control complexity	Moderate	High (excitation control)
Initial cost	Higher (rare-earth material)	Lower initial cost
Temperature sensitivity	Demagnetization	Less affected by temperature

Table 1.1: Comparison of PMSM and EESM [4]

Since the EESM does not rely on permanent magnets (rare-earth elements), its cost is lower than that of a PMSM for high power density applications [4], while maintaining acceptably high efficiency suitable for most applications. Another benefit of the EESM compared to other AC machines is the extra control degree of freedom offered by the excitation circuit in the rotor. This, however, increases the complexity of the power electronics and the control strategy employed.

1.1.2 Reinforcement-Learning-Based Control

RL is an inherently data-driven control method, meaning control actions are learned directly from data obtained through interactions with the controlled system, relying less on the system's plant modeling. Electrical drives are traditionally controlled using linear FOC techniques in conjunction with proportional–integral (PI) controllers. This approach is only slowly being replaced by more sophisticated methods of optimal control, especially MPC [7]. The latter requires higher computational effort during online operation and necessitates expert knowledge, either in the form of a prior model or through system identification techniques to derive a model from observed data, which is its primary drawback. MPC makes use of a mathematical model of the electrical drive, where typically some of the parasitic effects (e.g., inverter non-linearity, cross saturation, etc...) are neglected in order to reduce the computational burden [7]. Data-driven algorithms would allow the controller to learn and react to these effects, since they are implicitly included in the measurement data [7]. These considerations encourage the investigation of RL methods for AC drive control. Finally, there is already proof that PMSMs can be controlled using RL methods, as evidenced in [8] and [9].

1.2 Objective and Structure of Research

The primary objective of this thesis is to control the torque of an EESM using an RL agent. This entails developing a data-driven and highly adaptive controller that is applicable to this AC drive system. The optimization goal is to achieve minimum torque tracking error, minimum transient time, and optimal efficiency across a wide range of speeds, up to the base speed.

However, several challenges may arise, such as long training periods, balancing the exploration-exploitation dilemma, stabilizing the artificial neural network, and managing the storage requirements for extensive training data. This thesis aims to tackle these challenges, starting with the current control strategy to ensure that the RL agent can understand the complex environment and multi-dimensional state space. Once it is demonstrated that the agent can learn and handle tracking the reference current, the research will smoothly transition to optimal torque control in the following order.

Step: 1 Current control with RL agent

Step: 2 Torque control with PI-regulated rotor circuit and RL-regulated stator circuit

Step: 3 Torque control with RL agent

Step: 3.1 Performance-priority control

Step: 3.2 Efficiency-priority control

Successfully addressing these challenges would enable the design of an RL agent for optimal torque control, providing an alternative to issues associated with traditional methods. The analysis presented in this thesis will be conducted through simulations rather than experiments, as the primary objective is to provide an initial proof of concept demonstrating that complex systems like EESM can be effectively controlled using RL.

1.3 Conceptual Framework

As explained above, this thesis focuses on two main aspects: EESM as a drive system and RL as a controller. Firstly, the EESM will be discussed with the mathematical modelling and analytical representation of its working principle. Secondly, this section will introduce RL and its elements while addressing the challenges and objectives. It is important to note that the drive model given in the following is explained exclusively for contextual purposes in this report, and it is not known to the RL agent.

1.3.1 General EESM Modelling Approach

The derivation of the EESM model can be carried out on the basis of the schematic diagram as shown in figure 1-1. The rotor of the machine has one field winding, and saliency generally occurs due to its physical structure. However, the following approach is valid for both salient and non-salient EESM. In this thesis, damper windings are not considered. For the following mathematical expressions, the three-phase windings in the stator will be represented by coils a, b and c, each having an identical number of windings. Every coil of the machine is modeled as an ohmic-inductive load, leading to the machine's dynamic equation as follows:

$$\boldsymbol{u}_{\mathrm{abcf}} = \boldsymbol{R}_{\mathrm{abcf}}\boldsymbol{i}_{\mathrm{abcf}} + \frac{\mathrm{d}\,\boldsymbol{\Psi}_{\mathrm{abcf}}}{\mathrm{d}t}, \tag{1.1}$$

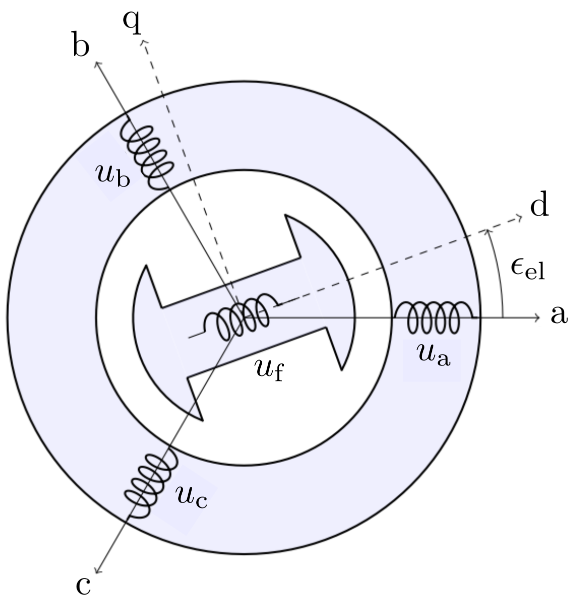


Figure 1-1: Schematic representation of EESM [1]

where $\boldsymbol{u}_\mathrm{abcf} = [u_\mathrm{a}, u_\mathrm{b}, u_\mathrm{c}, u_\mathrm{f}]^\mathrm{T}$ is the vector of phase voltages and excitation voltage, $\boldsymbol{i}_\mathrm{abcf} = [i_\mathrm{a}, i_\mathrm{b}, i_\mathrm{c}, i_\mathrm{f}]^\mathrm{T}$ is the vector of phase currents and excitation current, $\boldsymbol{\Psi}_\mathrm{abcf} = [\Psi_\mathrm{a}, \Psi_\mathrm{b}, \Psi_\mathrm{c}, \Psi_\mathrm{f}]^\mathrm{T}$ is the vector of stator flux linkage and excitation flux linkage and $\boldsymbol{R}_\mathrm{abcf} = \mathrm{diag}([R_\mathrm{s}, R_\mathrm{s}, R_\mathrm{s}, R_\mathrm{f}])$ is the diagonal matrix of ohmic resistance.

For the derivation of the motor model, the parameter dependencies play an important role and will be discussed below. Generally, the flux linkage depends on the stator current, excitation current and rotor angle (ϵ_el) [10]. The dynamics of temperature changing will be neglected in this mathematical modelling of machine. Therefore, it can be described as follow:

$$\boldsymbol{\Psi}_\mathrm{abcf} = f(i_\mathrm{a}, i_\mathrm{b}, i_\mathrm{c}, i_\mathrm{f}, \epsilon_\mathrm{el}). \tag{1.2}$$

For the following simplified model, only linear magnetization is considered as

$$\boldsymbol{\Psi}_\mathrm{abcf} = \boldsymbol{L}_\mathrm{abcf}\boldsymbol{i}_\mathrm{abcf}, \tag{1.3}$$

where $\boldsymbol{L}_{\mathrm{abcf}}$ is the absolute inductances which vary with respect to the rotor angle. It can be seen as a matrix:

$$\boldsymbol{L}_{\mathrm{abcf}} = \begin{bmatrix} L_{\mathrm{aa}} & M_{\mathrm{ab}} & M_{\mathrm{ac}} & M_{\mathrm{af}} \\ M_{\mathrm{ba}} & L_{\mathrm{bb}} & M_{\mathrm{bc}} & M_{\mathrm{bf}} \\ M_{\mathrm{ca}} & M_{\mathrm{cb}} & L_{\mathrm{cc}} & M_{\mathrm{cf}} \\ M_{\mathrm{fa}} & M_{\mathrm{fb}} & M_{\mathrm{fc}} & L_{\mathrm{ff}} \end{bmatrix}, \tag{1.4}$$

whereas $L_{ii}(i \in \{\mathrm{a,b,c,f}\})$ are the time-variant absolute self inductances and $M_{ij}(i,j \in \{\mathrm{a,b,c,f}\} \wedge i \neq j)$ are the time-variant absolute mutual inductances [10].

Reference Frame Transformations

For the analysis of electrical machines, especially when dealing with 3-phase AC machines, phase currents $(i_{\mathrm{a}}, i_{\mathrm{b}}, i_{\mathrm{c}})$ come in sinusoidal or time-variant waveform, making the control structure complicated. To simplify the control of AC drives, an axis transformation is performed. This involves transforming the 3-phase stator to a fictitious 2-phase stator, and then from the 2-phase stator to a rotating dq frame. Direct transformation from the 3-phase stator to the rotating dq frame is also possible.

$$\begin{bmatrix} x_{\alpha} \\ x_{\beta} \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} x_{\mathrm{a}} \\ x_{\mathrm{b}} \\ x_{\mathrm{c}} \end{bmatrix} = \boldsymbol{T}_{\mathrm{abc}\to\alpha\beta}\ \boldsymbol{x}_{\mathrm{abc}}, \tag{1.5}$$

$$\boldsymbol{T}_{\alpha\beta\to\mathrm{abc}} = \mathbf{T}^{\dagger}_{\mathrm{abc}\to\alpha\beta} = \frac{3}{2} \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{\sqrt{3}}{3} & -\frac{\sqrt{3}}{3} \end{bmatrix}^{\mathrm{T}}. \tag{1.6}$$

There, † depicts the matrix pseudo-inverse (for non-square matrices). The notation x is used for general purpose and the scaling of $\frac{2}{3}$ is used to get the same amplitude for the variables in the equivalent 2-phase model. When dealing with the EESM, the excitation variable is also taken into consideration.

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_\mathrm{f} \end{bmatrix} = \begin{bmatrix} \boldsymbol{T}_{\mathrm{abc}\to\alpha\beta} & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} x_\mathrm{a} \\ x_\mathrm{b} \\ x_\mathrm{c} \\ x_\mathrm{f} \end{bmatrix} = \boldsymbol{T}_{\mathrm{abcf}\to\alpha\beta\mathrm{f}}\, \boldsymbol{x}_\mathrm{abcf}, \tag{1.7}$$

$$\boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{abcf}} = \mathbf{T}^\dagger_{\mathrm{abcf}\to\alpha\beta\mathrm{f}} = \begin{bmatrix} \boldsymbol{T}_{\alpha\beta\to\mathrm{abc}} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 2} & 1 \end{bmatrix}. \tag{1.8}$$

To use a synchronous reference frame to the rotor, the rotor field oriented coordinates transformation is done as follow:

$$\begin{bmatrix} x_\mathrm{d} \\ x_\mathrm{q} \end{bmatrix} = \begin{bmatrix} \cos(\epsilon_\mathrm{el}) & \sin(\epsilon_\mathrm{el}) \\ -\sin(\epsilon_\mathrm{el}) & \cos(\epsilon_\mathrm{el}) \end{bmatrix} \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = \boldsymbol{T}_{\alpha\beta\to\mathrm{dq}}\, \boldsymbol{x}_{\alpha\beta}, \tag{1.9}$$

$$\boldsymbol{T}_{\mathrm{dq}\to\alpha\beta} = \boldsymbol{T}^{-1}_{\alpha\beta\to\mathrm{dq}} = \begin{bmatrix} \cos(\epsilon_\mathrm{el}) & -\sin(\epsilon_\mathrm{el}) \\ \sin(\epsilon_\mathrm{el}) & \cos(\epsilon_\mathrm{el}) \end{bmatrix}, \tag{1.10}$$

$$\begin{bmatrix} x_\mathrm{d} \\ x_\mathrm{q} \\ x_\mathrm{f} \end{bmatrix} = \begin{bmatrix} \cos(\epsilon_\mathrm{el}) & \sin(\epsilon_\mathrm{el}) & 0 \\ -\sin(\epsilon_\mathrm{el}) & \cos(\epsilon_\mathrm{el}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\alpha \\ x_\beta \\ x_\mathrm{f} \end{bmatrix} = \boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{dqf}}\, \boldsymbol{x}_{\alpha\beta\mathrm{f}}, \tag{1.11}$$

$$\boldsymbol{T}_{\mathrm{dqf}\to\alpha\beta\mathrm{f}} = \boldsymbol{T}^{-1}_{\alpha\beta\mathrm{f}\to\mathrm{dqf}} = \begin{bmatrix} \boldsymbol{T}_{\mathrm{dq}\to\alpha\beta} & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 2} & 1 \end{bmatrix}, \tag{1.12}$$

where ϵ_el is the rotor's electrical angle which is changing with respect to time. With the help of above-mentioned axes transformation, the dynamic equation (1.1) is rewritten in $\alpha\beta$ reference frame as follow:

$$\boldsymbol{u}_{\alpha\beta\mathrm{f}} = \boldsymbol{R}_{\alpha\beta\mathrm{f}} \boldsymbol{i}_{\alpha\beta\mathrm{f}} + \frac{\mathrm{d}\boldsymbol{\Psi}_{\alpha\beta\mathrm{f}}}{\mathrm{d}t}, \tag{1.13}$$

where

$$\boldsymbol{R}_{\alpha\beta\mathrm{f}} = \boldsymbol{T}_{\mathrm{abcf}\to\alpha\beta\mathrm{f}} \boldsymbol{R}_\mathrm{abcf} \boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{abcf}}, \tag{1.14}$$

and in the dq reference frame,

$$\boldsymbol{u}_{\mathrm{dqf}} = \boldsymbol{R}_{\mathrm{dqf}}\boldsymbol{i}_{\mathrm{dqf}} + \frac{\mathrm{d}\,\boldsymbol{\Psi}_{\mathrm{dqf}}}{\mathrm{d}t} + \omega_{\mathrm{el}}\boldsymbol{J}\,\boldsymbol{\Psi}_{\mathrm{dqf}}, \tag{1.15}$$

where

$$\boldsymbol{R}_{\mathrm{dqf}} = \boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{dqf}}\boldsymbol{R}_{\alpha\beta\mathrm{f}}\boldsymbol{T}_{\mathrm{dqf}\to\alpha\beta\mathrm{f}}. \tag{1.16}$$

In (1.15), $\boldsymbol{J}$ is the result of mathematical operation of axis transformation which can be seen as:

$$J = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{1.17}$$

Then, ω_{el} from equation (1.15) is the electrical speed of the machine and it can be obtained from a rate of change of electrical position with respect to time as follows:

$$\omega_{\mathrm{el}} = \frac{\mathrm{d}\epsilon_{\mathrm{el}}}{\mathrm{d}t} \quad \text{and} \quad \omega_{\mathrm{el}} = p\omega_{\mathrm{mech}}, \tag{1.18}$$

where ω_{mech} is the mechanical speed of the machine and p is the number of pole-pairs. Therefore, the above dynamic equation can be rewritten as below:

$$u_{\mathrm{sd}} = R_{\mathrm{s}}i_{\mathrm{sd}} + \dot{\Psi}_{\mathrm{sd}} - \omega_{\mathrm{el}}\,\Psi_{\mathrm{sq}}, \tag{1.19a}$$

$$u_{\mathrm{sq}} = R_{\mathrm{s}}i_{\mathrm{sq}} + \dot{\Psi}_{\mathrm{sq}} + \omega_{\mathrm{el}}\,\Psi_{\mathrm{sd}}, \tag{1.19b}$$

$$u_{\mathrm{f}} = R_{\mathrm{f}}i_{\mathrm{f}} + \dot{\Psi}_{\mathrm{f}}. \tag{1.19c}$$

The dq reference frame is defined with the d-axis which is aligned with the rotor's axis. After applying the appropriate transformation to equation (1.3), the flux equation results in

$$\boldsymbol{\Psi}_{\mathrm{dqf}} = \boldsymbol{L}_{\mathrm{dqf}}\boldsymbol{i}_{\mathrm{dqf}}, \tag{1.20}$$

where

$$\boldsymbol{L}_{\mathrm{dqf}} = \boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{dqf}}\boldsymbol{L}_{\alpha\beta\mathrm{f}}\boldsymbol{T}_{\mathrm{dqf}\to\alpha\beta\mathrm{f}}, \tag{1.21}$$

$$\boldsymbol{L}_{\alpha\beta\mathrm{f}} = \boldsymbol{T}_{\mathrm{abcf}\to\alpha\beta\mathrm{f}}\boldsymbol{L}_{\mathrm{abcf}}\boldsymbol{T}_{\alpha\beta\mathrm{f}\to\mathrm{abcf}}. \tag{1.22}$$

Then, the equation (1.20) can be rewritten as:

$$\Psi_{\mathrm{sd}} = L_{\mathrm{d}}i_{\mathrm{sd}} + L_{\mathrm{m}}i_{\mathrm{f}}, \tag{1.23a}$$

$$\Psi_{\mathrm{sq}} = L_{\mathrm{q}}i_{\mathrm{sq}}, \tag{1.23b}$$

$$\Psi_{\mathrm{f}} = L_{\mathrm{f}}i_{\mathrm{f}} + \frac{3}{2}L_{\mathrm{m}}i_{\mathrm{sd}}, \tag{1.23c}$$

which exhibits significant cross-coupling between the d- and f-axis. The inductance in matrix form can be seen as follow [5]:

$$\boldsymbol{L}_{\mathrm{dqf}} = \begin{bmatrix} L_{\mathrm{d}} & 0 & L_{\mathrm{m}} \\ 0 & L_{\mathrm{q}} & 0 \\ \frac{3}{2}L_{\mathrm{m}} & 0 & L_{\mathrm{f}} \end{bmatrix}. \tag{1.24}$$

The $\frac{3}{2}$ term appears as a consequence of the coordinate transformation criteria provided in equation (1.5) to (1.12).

1.3.2 Development of Electromagnetic Torque

The electrical machines are electro-mechanical energy converters, and transform electrical energy into mechanical energy or vice versa according to the operating regions. The electromagnetic torque is developed through the interaction of magnetic fields established by the currents in the stator and rotor, and it can be expressed as follows [11]:

$$T_{\mathrm{em}} = \frac{3}{2}p\left(-i_{\mathrm{sd}}\,\Psi_{\mathrm{sq}} + i_{\mathrm{sq}}\,\Psi_{\mathrm{sd}}\right). \tag{1.25}$$

By substituting the equations (1.23a) and (1.23b) into the above equation, torque can be rewritten as:

$$\begin{aligned} T_{\mathrm{em}} &= \frac{3}{2}p\Big(-i_{\mathrm{sd}}(L_{\mathrm{q}}i_{\mathrm{sq}}) + i_{\mathrm{sq}}(L_{\mathrm{d}}i_{\mathrm{sd}} + L_{\mathrm{m}}i_{\mathrm{f}})\Big), \\ T_{\mathrm{em}} &= \frac{3}{2}p\Big(\underbrace{L_{\mathrm{m}}i_{\mathrm{f}}i_{\mathrm{sq}}}_{\text{main torque}} + \underbrace{(L_{\mathrm{d}} - L_{\mathrm{q}})i_{\mathrm{sd}}i_{\mathrm{sq}}}_{\text{reluctance torque}}\Big). \end{aligned} \tag{1.26}$$

The electromagnetic torque can be divided into two parts: the main torque which is generated by the interaction between the rotor current and the stator current, and the reluctance torque which arises from the difference in inductances along the d-axis and q-axis of the machine.

It is important to highlight the behaviour of d- and q- axis inductances in EESM because it is quite different from PMSM. In surface PMSM, if the magnet is mounted on the surface of the rotor, $L_{\mathrm{d}} = L_{\mathrm{q}}$ and no reluctance torque will be developed. In interior PMSM, if the magnet is put inside the rotor, under the consideration of rotor flux orientation, $L_{\mathrm{d}} < L_{\mathrm{q}}$ because of the magnet's permeability. Therefore, in interior PMSM, negative d-axis current is preferred in order to have positive reluctance torque. However, the scenario of $L_{\mathrm{d}} > L_{\mathrm{q}}$ is not common in PMSM while it is normal in EESM.

In EESM, the values of L_{d} and L_{q} mainly depend on rotor geometry, and due to the rotor winding slots, d-axis inductance is typically higher than q-axis inductance [10]. In cases of high magnetic saturation, the situation may reverse, leading to $L_{\mathrm{d}} < L_{\mathrm{q}}$. This scenario, however, is not considered in the scope of this research

1.3.3 Current and Voltage Limit

The stator current is limited by the thermal characteristics of the input electrical power supply and the motor. The maximum available current for d- and q-axes can be defined as follow:

$$i_{\mathrm{sd}}^2 + i_{\mathrm{sq}}^2 = i_{\mathrm{s}}^2 \leq I_{\mathrm{s,max}}^2. \tag{1.27}$$

Similarly, the stator voltage is limited by the voltage output capability of the power electronic converter and the insulation level of the motor winding. The maximum available voltage for d- and q-axes is:

$$u_{\mathrm{sd}}^2 + u_{\mathrm{sq}}^2 \leq U_{\mathrm{s,max}}^2. \tag{1.28}$$

If the space vector modulation or homo-polar harmonic injection is used for switching pattern, the maximum voltage in the linear modulation region becomes:

$$U_{\mathrm{s,max}} = \frac{U_{\mathrm{DC}}}{\sqrt{3}}. \tag{1.29}$$

1.3.4 Optimal Operation of EESMs

The maximum torque per current (MTPC) is one of the popular methods to operate the AC drive when the motor is running below the base speed, because copper losses are minimized and maximum efficiency is achieved. Thanks to the field current, there is one additional degree of freedom for EESM compared to PMSM. However, this means that more than one current contributes to reduce copper losses. Therefore, instead of MTPC, maximum torque per copper losses (MTPCL) is more appropriate for operating EESM. Thus, the term MTPCL will be utilized in the following sections. using the parameters from table 1.2.

The behaviors of i_{sd}, i_{sq}, i_{f} and i_{stator} are illustrated in figure 1-2, using the parameters from table 1.2. This analysis covers operating points ranging from minimal to maximum torque under the MTPCL method, for steady-state conditions with only current limitations applied.

1.3.5 Power Electronic Converters Modelling

A fundamental component of electric drives are the power electronic converters, which apply the desired voltage actions to the motor terminals. For the two main circuits involved in EESM (the excitation circuit for the rotor and the armature circuit for the stator), at least two power electronic converters are required.

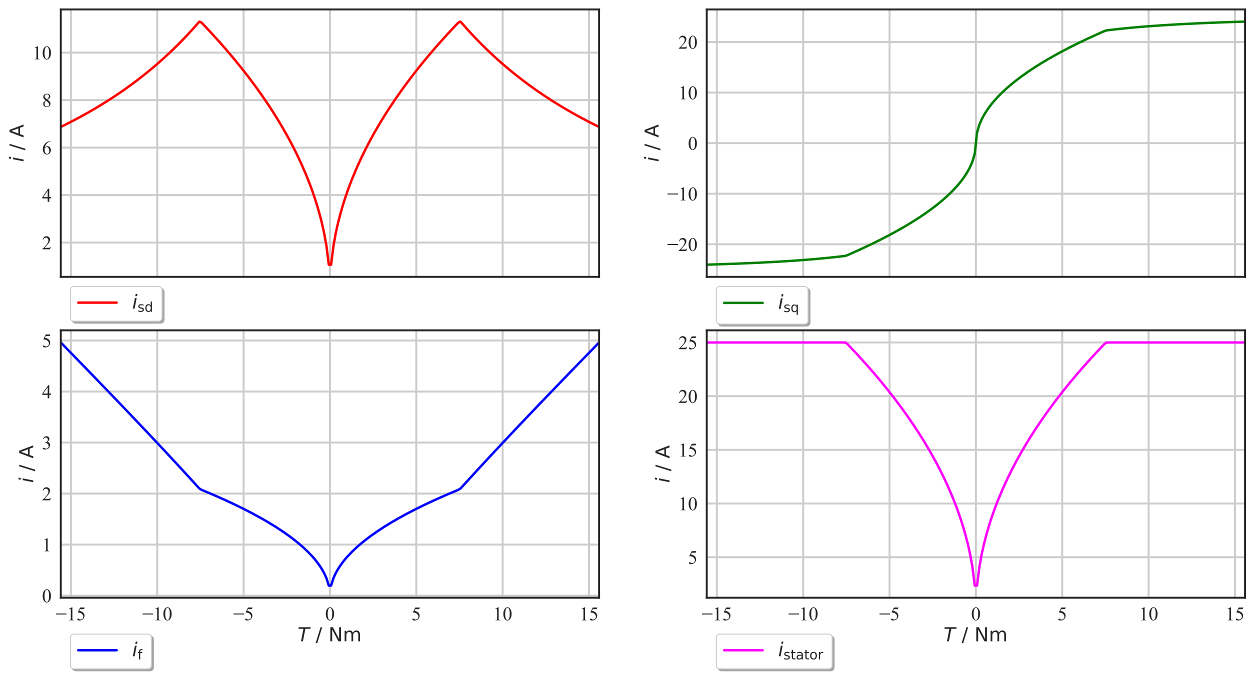


Figure 1-2: MTPCL current trajectories

Four Quadrant Converter

For the rotor circuit, a DC/DC converter is required to apply (in average) arbitrary voltages to the excitation circuit terminals from the available DC bus. In this thesis, the 4-quadrant converter (4QC) is considered as shown in figure 1-3.

The 4QC is versatile and applicable to different voltage outputs ranging from $+U_{\text{DC}}$ to $-U_{\text{DC}}$. For this use-case, quadrants 1 and 4 are utilized since it is only necessary to control the excitation current from zero to its upper limit based on the operating points of EESM. According to the converter's structure, the available excitation voltage would be $u_{\text{f}} \in [-U_{\text{DC}}, U_{\text{DC}}]$. From the controller perspective, the action command provided is the desired voltage u_{f}^{*}. This voltage command is, in a real application, achieved by appropriate switching pulses generated by a modulation technique, such as PWM or SVM. In this analysis, the switching pulses are not considered, and the converter model will apply the desired (limited) average voltage for the duration of the sampling period.

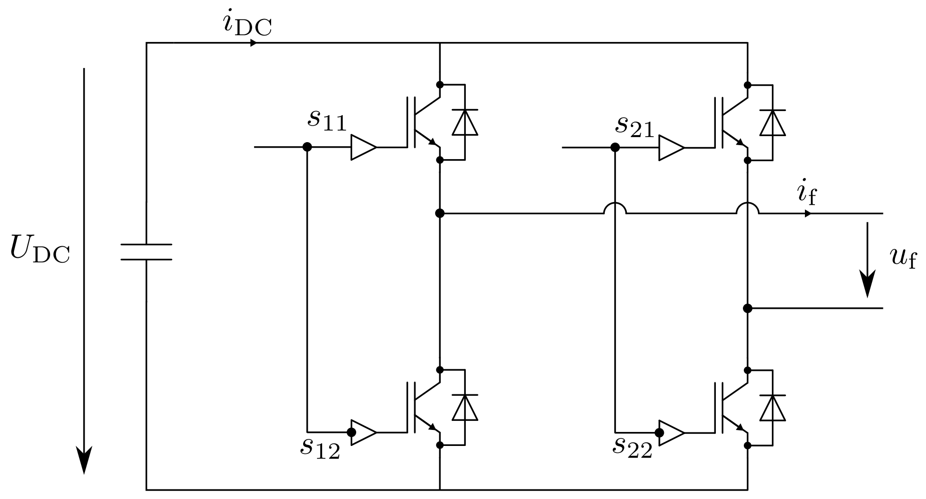


Figure 1-3: Four quadrant converter

B6 Bridge Converter

For the control of the stator circuit of three-phase EESM, a three-phase two-level converter or B6 bridge three-phase converter is considered as shown in figure 1-4.

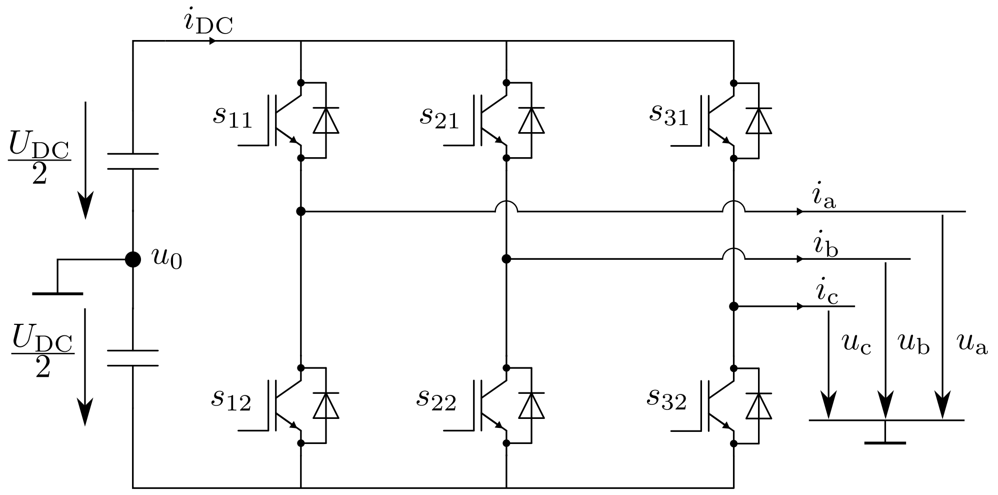


Figure 1-4: B6 bridge converter

The inputs to the B6 bridge converter are the supply DC link voltage comes from the battery or rectified grid supply, and the switching commands for the transistors (IGBTs/

MOSFETs) to turn ON or OFF. Typically, the controllers provide the desired output voltage which needs to be mapped to a switching pattern over time. The switching schemes determine the output voltage of the converter, with various modulation methods available for this purpose. Pulse width modulation (PWM) with homo-polar harmonics injection and space vector modulation (SVM) are very common. As seen in figure, the available stator voltage is $u_\mathrm{s} \in \left[-\frac{U_\mathrm{DC}}{2}, \frac{U_\mathrm{DC}}{2}\right]$. As mentioned above, the action command provided is the desired voltage $\boldsymbol{u}^*_\mathrm{dq}$. The converter model applies this desired (limited) average voltage over the sampling period, without considering the switching pulses.

1.3.6 Gym Electric Motor Toolbox

The gym-electric-motor (GEM) package is a software toolbox in python for the simulation of different electric motors to train and test RL motor controllers, and it models an electric drive system by its four main components: voltage supply, mechanical load, converter and electric motor [12]. The general structure of such a system is depicted in figure 1-5.

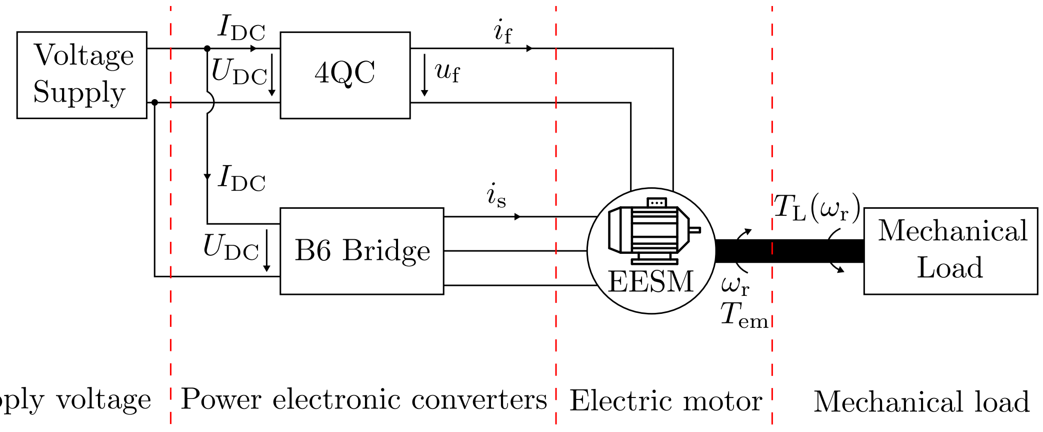


Figure 1-5: Simplified structure diagram of an electric drive system

The *voltage supply* is modeled by a fixed supply voltage with $U_\mathrm{DC} = 200\,\mathrm{V}$ as per the selected motor parameters, providing the necessary power to run the motor. Then, for the *mechanical load*, the constant speed load is considered in this report.

For the *power electronic converters*, GEM offers different converters and as mentioned above, 4QC and B6 bridge converters are considered in this thesis for rotor and stator circuits. The switching states determine the output voltage of the converter, however, from a simulation point of view, these methods require very tiny time steps to accurately cover the switching instants. Therefore, to speed up the simulation and reduce the computer's CPU requirements, the modulation schemes are neglected, and a dynamic average model is used in the GEM environment [13].

For the *electric motor*, a variety of models are available in the GEM toolbox, with EESM being used in this thesis as discussed in the motivation. In RL terms, the EESM and the converters will act as the environment, since the RL controller interacts with the electric drive and learns the system over time. Therefore, later in this report, the term "environment" refers to the drive system of the EESM. In addition, GEM provides both continuous control set (CCS) and finite control set (FCS) options for controlling the current, speed, and torque of the EESM. However, this thesis will focus on CCS, and the relevant environment will be used accordingly in the following sections.

1.3.7 Introduction to Reinforcement Learning and its Elements

As illustrated in figure 1-6, RL is a branch of machine learning (ML), specifically designed for decision-making processes. The general tasks associated with each branch of ML are shown in table 1.3.

The elements of RL are depicted in figure 1-7, with the *agent* and *environment* as the fundamental components. The environment represents the system or entity that the agent interacts with [17]—in this case, the EESM drive. The agent is the entity that interacts with the environment to achieve specific goals. In this use case, the agent functions as the controller, making decisions to control the drive system using a predefined action set. Throughout this report, the term "agent" refers to the controller.

Symbol	Description	Value	Unit
p	number of pole-pairs	2	-
L_d	d-axis inductance	3.78	mH
L_q	q-axis inductance	1.21	mH
L_f	rotor inductance	870	mH
L_m	coupling inductance	40	mH
R_s	stator resistance	123	mΩ
R_f	rotor resistance	15.6	Ω
n	winding ratio $\frac{N_\mathrm{s}}{N_\mathrm{f}}$	0.057	-
$i_\mathrm{s,nom}$	nominal stator current	25	A
$i_\mathrm{f,nom}$	nominal rotor current	5	A
u_DC	DC link voltage	200	V
$\omega_\mathrm{r,lim}$	maximum angular velocity	7000	$\frac{1}{\mathrm{min}}$
$\omega_\mathrm{r,base}$	base angular velocity	2000	$\frac{1}{\mathrm{min}}$
T_nom	nominal torque	15	Nm

Table 1.2: Parameterization of the considered drive system, EESM [5]

Markov Decision Processes

RL is generally used to solve Markov decision processes (MDPs), which are a mathematically idealized form of RL problems. MDPs allow for precise theoretical statements and provide insights into RL solutions, as many real-world problems can be abstracted as MDPs [3]. Therefore, MDPs are discussed below, and it is important to note that this thesis focuses exclusively on MDPs where all states are fully measurable.

An MDP is defined by a tuple $(\mathcal{X}, \mathcal{U}, \boldsymbol{\mathcal{P}}, \mathcal{R}, \gamma)$, where $\mathcal{X}$ is a set of states. In this investigation, $\mathcal{X} = \{\omega_\mathrm{mech}, T, i_\mathrm{sd}, i_\mathrm{sq}, i_\mathrm{f}, \epsilon_\mathrm{el}, ...\}$ are general representation of states of EESM as environment regardless of time step and fully observable to the agent. Then, $\mathcal{U}$ is a set of discrete-time actions $U_\mathrm{k} \in \mathcal{U}$. As discussed in the EESM section, the d-, q- and f-axes voltages are involved in the dynamics equations (1.19) and are considered actions in the RL

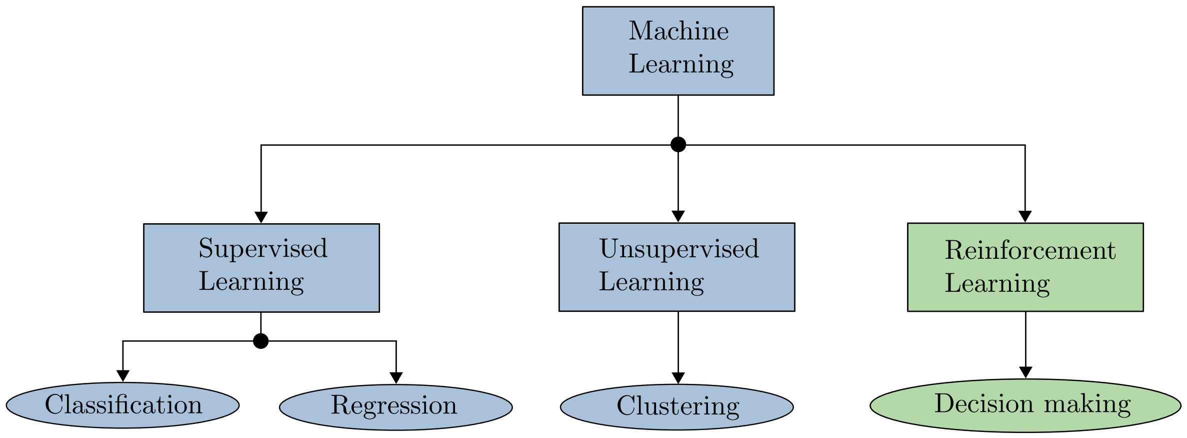


Figure 1-6: General disciplines of machine learning

Supervised learning	Learning a mapping between a set of input variables and output variables, then applying this mapping to predict the outputs for unseen data [14].
Unsupervised learning	Learning to represent particular input patterns in a way that reflects the statistical structure of the overall collection of input patterns [15].
Reinforcement learning	Learning over time to provide optimal control actions by interacting with an environment [16].

Table 1.3: General purposes of different branches of machine learning

framework. Therefore, the action set in this analysis is $\mathcal{U} = \{u_{\text{sd}}, u_{\text{sq}}, u_{\text{f}}\}$. Moreover, $\boldsymbol{\mathcal{P}}$, the state transition probability is defined as:

$$\boldsymbol{\mathcal{P}} = \boldsymbol{\mathcal{P}}_{xx'} = \mathbb{P}\left[X_{k+1} = x' \mid X_k = x\right], \tag{1.30}$$

which means the transition probability of moving from state x to x' with respect to time step, and in this analysis, a deterministic environment is considered, where the state transition is governed by the EESM's ordinary differential equations (ODEs) as outlined in the preceding sections.

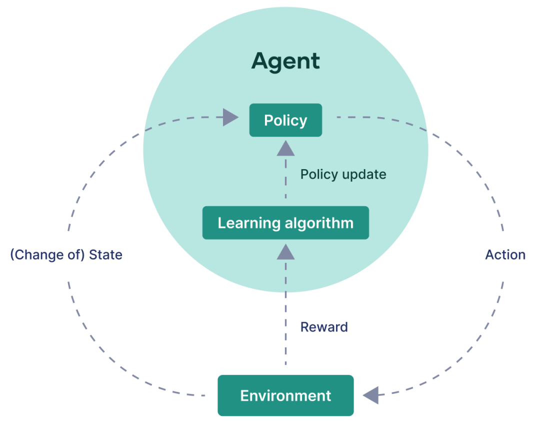


Figure 1-7: The basic RL operation principle [2]

In addition, $\mathcal{R}$ is a reward function which is defined mathematically as:

$$\mathcal{R} = \mathcal{R}_x^u = \mathbb{E}\left[R_{k+1} \mid X_k = x_k, U_k = u_k\right]. \tag{1.31}$$

This represents the expected reward received after taking action $\mathcal{U}$ in state $\mathcal{X}$ and therefore, the agent receives a reward on each time step. While the reward directs the agent to control the EESM in the desired behaviour, the return (the cumulative reward over an episode or over time) as shown in equation (1.32) is more significant when evaluating how well the agent performs throughout the episode or across the duration of a continuing task.

In RL, the interactions between an agent and its environment are typically structured into episodes. An episode is a sequence of states, actions, rewards and state transitions that the agent experiences. There are generally two types of episodes: one with a termination state and the other without a natural end. For example, winning a chess game represents a termination state, marking the end of the episode, whereas controlling AC drives is a continuing task without a specific endpoint. For an episodic task, the return can be defined in mathematical form as:

$$g_k = r_{k+1} + r_{k+2} + ... + r_N, \tag{1.32}$$

where g_k is the return at time step k which is the summation of the rewards from time step

$k+1$ until the episode end, terminal step $k = N$ [3]. For a continuing task, the future rewards should be discounted to prevent the return becoming infinite. That is why, γ, the discount factor is involved in a tuple of MDP, and $\gamma \in \{\mathbb{R} \mid 0 \leq \gamma \leq 1\}$. Then, the return becomes

$$g_k = r_{k+1} + \gamma r_{k+2} + \gamma^2 r_{k+3} + ... = \sum_{i=0}^{\infty} \gamma^i r_{k+i+1}. \tag{1.33}$$

The return can also be calculated recursively as follows:

$$g_k = r_{k+1} + \gamma \, g_{k+1}. \tag{1.34}$$

Concerning the range of discount rate, the return can be discussed in numeric viewpoint and strategic viewpoint [3].

Numeric Viewpoint:

If $\gamma = 1$ and $r_k > 0$ for $k \longrightarrow \infty$, g_k in equation (1.33) becomes infinity.

If $\gamma < 1$ and r_k is bounded for $k \longrightarrow \infty$, g_k in equation (1.33) is bounded which means the return would be within a certain range.

Strategic Viewpoint:

If $\gamma \approx 1$, the agent is farsighted, looking forward to the future reward.

If $\gamma \approx 0$, the agent is shortsighted, only interested in immediate reward.

Then, if $r_k = r$ is a constant and $\gamma < 1$, the return from equation (1.33) would be [3]:

$$g_k = \sum_{i=0}^{\infty} \gamma^i r = r \sum_{i=0}^{\infty} \gamma^i = r \frac{1}{1-\gamma}. \tag{1.35}$$

Therefore, the returns can be used to evaluate how good the agent's actions are over the long run, not just in the immediate future.

The MDP has the Markov property, which implies that the future state is dependent solely on the current state and action, not on the sequence of past states [17]. Mathematically, this is expressed as:

$$\mathbb{P}(X_{k+1} = x' \mid X_k = x) = \mathbb{P}(X_{k+1} = x' \mid X_0 = x_0, X_1 = x_1, \ldots, X_k = x). \tag{1.36}$$

This equation highlights the memory-less property, a key aspect of the Markov property, indicating that the transition to the next state X_{k+1} depends only on the current state X_k, and not on any previous states. Furthermore, if the state transition probabilities are consistent over time, the process is said to have the time-homogeneous property [17]. This is expressed as:

$$\mathbb{P}(X_{k+2} = x' \mid X_{k+1} = x) = \mathbb{P}(X_{k+1} = x' \mid X_k = x), \tag{1.37}$$

for any time step k and all possible states. The time-homogeneous property ensures that transition probabilities remain constant over time, meaning the rules governing state transitions do not change. As a result, the MDP's state transitions depend solely on the current state and action, reflecting both the memory-less (Markov) and time-homogeneous properties.

The objective of the MDP is to find the optimal policy π^* that maximizes the return. A policy is a distribution over actions given states [3]. It could be a strategy or a rule that guides an agent in selecting specific actions for each state over time, as illustrated in equation (1.39), making it an integral part of the agent. Generally, the policy can be described as:

$$\pi(u_k \mid x_k) = \mathbb{P}\left[U_k = u_k \mid X_k = x_k\right]. \tag{1.38}$$

In MDPs, policies can be either stochastic or deterministic, and they depend solely on the current state, fully defining the agent's behavior [3]. The deterministic policy is intended to be used in this analysis (detailed discussion in subsection 1.3.8), and it involves directly mapping states to actions:

$$\boldsymbol{u}_k = \boldsymbol{\pi}(\boldsymbol{x}_k). \tag{1.39}$$

For example, when someone is driving a car, a policy might be a set of rules like "$\underline{\text{if the traffic light is red}}$, stop and $\underline{\text{if it's green}}$, go", where *stopping* and *going* are the actions and the underlined phrases represent the policies. In some cases the policy is a simple function or lookup table, while in complex cases, it may involve the extensive computation such as a search process [16].

The concept of a value function is introduced to determine how good a particular policy is. A value function represents the estimated expected return, helping the agent evaluate the desirability of states or state-action pairs under a particular policy. Generally, there are two types of value functions: the state-value function and the action-value function. However, for the utilization of a deterministic policy, the action-value function is of greater interest.

The action-value function of an MDP is the expected return starting in state x_k taking the action u_k and then following the policy π:

$$q_\pi(x_k, u_k) = \mathbb{E}_\pi[G_k | X_k = x_k, U_k = u_k] = \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i r_{k+i+1} | x_k, u_k\right]. \tag{1.40}$$

For example, if $q_\pi(x_k, u_k) = 0.9$, it means that the expected return from state x_k, taking the action u_k, and afterwards following the policy π is 0.9. In order to calculate all the action values in a closed form, the Bellman equation can be used where the value function is recursively defined. The above equation (1.40) can be written as:

$$\begin{aligned} q_\pi(x_k, u_k) &= \mathbb{E}_\pi\big[R_{k+1} + \gamma R_{k+2} + \gamma^2 R_{k+3} + ... | X_k = x_k, U_k = u_k\big] \\ &= \mathbb{E}_\pi\big[R_{k+1} + \gamma(R_{k+2} + \gamma R_{k+3} + ...) | X_k = x_k, U_k = u_k\big] \\ &= \mathbb{E}_\pi\big[R_{k+1} + \gamma G_{k+1} | X_k = x_k, U_k = u_k\big] \\ &= \mathbb{E}_\pi\big[R_{k+1} + \gamma q_\pi(X_{k+1}, U_{k+1}) | X_k = x_k, U_k = u_k\big]. \end{aligned} \tag{1.41}$$

Therefore, the action-value function can be mentioned as Bellman equation incorporating transition probabilities explicitly as follow:

$$q_\pi(x_k, u_k) = \mathcal{R}_{x_k}^{u_k} + \gamma \sum_{x_{k+1} \in \mathcal{X}} p_{x_k x_{k+1}}^{u_k} \left(\sum_{u_{k+1} \in \mathcal{U}} \pi(u_{k+1} | x_{k+1}) q_\pi(x_{k+1}, u_{k+1}) \right), \tag{1.42}$$

where $p_{x_k x_{k+1}}^{u_k}$ is the transition probability from state x_k to x_{k+1} under the action u_k. As mentioned above, the goal of an MDP is to find the optimal policy, which is a policy that is better than or equal to all other policies. Although there may be more than one [16], all optimal policies are denoted as π^*. They share the same optimal action-value function, denoted as q^*. The optimal action-value function of an MDP is the maximum action-value function over all polices:

$$q^*(x, u) = \max_{\pi} q_\pi(x, u). \tag{1.43}$$

The Bellman optimality equation is applicable to the action value and for the finite MDP,

$$q^*(x_k, u_k) = \mathcal{R}_{x_k}^{u_k} + \gamma \sum_{x_{k+1} \in \mathcal{X}} p_{x_k x_{k+1}}^{u_k} \max_{u_{k+1}} q^*(x_{k+1}, u_{k+1}). \tag{1.44}$$

1.3.8 Introduction to Deep Deterministic Policy Gradient

Deep RL will be used to control the EESM drive. Among the deep RL algorithms, the deep deterministic policy gradient (DDPG) algorithm [18] is selected for this thesis, as continuous action spaces are considered for the machine, and a deterministic policy is preferred for drive control. The DDPG algorithm was inspired by the success of Deep Q-Networks (DQNs) in discrete action spaces [19]. However, DQNs are not directly applicable to continuous action spaces, leading to the development of DDPG, which combines DQNs and Deterministic Policy Gradient (DPG) methods. The applications of DQNs and DDPG algorithms in controlling PMSM drives are discussed in [8] and [9]. In DDPG, an actor-critic off-policy approach is used to handle continuous state and action spaces.

Deterministic Policy Gradient

The DPG algorithm is a policy gradient method which works with deterministic policies instead of stochastic policies. Policy gradient methods optimize the policy directly by computing the gradients of the expected return with respect to the policy parameters. The objective is to maximize the expected return $J(\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the parameters of the policy. The DPG only integrates over the state space for the policy as mentioned in equation (1.39), while the stochastic case integrates over both state and action spaces for the policy gradient.

In other words, deterministic policies directly map states to actions while the action is drawn from a probability distribution in stochastic policies [20]. As the name stated, the DPG emphasizes on deterministic parameterized policies $\pi(x, \theta)$, and the gradient of $J(\theta)$ is defined as [3]:

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi \left[\nabla_\theta \pi(x,\theta) \nabla_u q(x,u) \Big|_{u=\pi(x)} \right]. \tag{1.45}$$

As seen above, the equation includes two gradients:

- $\nabla_\theta \pi(x,\theta)$: policy gradient with respect to its parameters,
- $\nabla_u q(x,u)$: gradient of the action-value function with respect to the action.

Artificial Neural Network

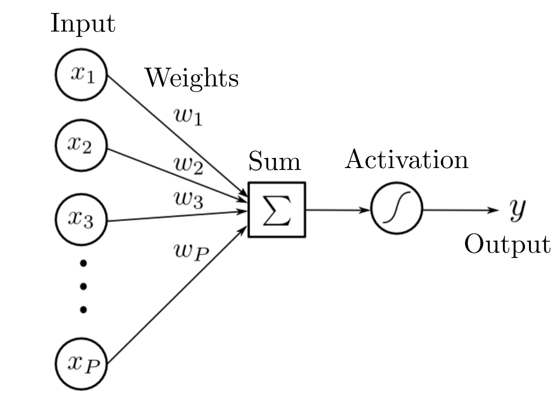
As previously discussed, DDPG is well-suited for environments with continuous action spaces and employs an actor-critic architecture. In this architecture, with the help of artificial neural networks (ANNs), the actor network determines the actions, while the critic network evaluates these actions.

ANNs are fundamental components of many modern machine learning algorithms, including those in reinforcement learning. Inspired by the structure and function of the human brain, ANNs consist of an input layer, an output layer, and one or more hidden layers. These layers contain neurons or nodes that are interconnected. Each neuron in the network processes the weighted sum of inputs from the previous layer and applies an activation function to produce an output, as illustrated in figure 1-8. The connections between neurons are represented by edges, which correspond to the parameters of the network. The output of a neuron can be mathematically represented as:

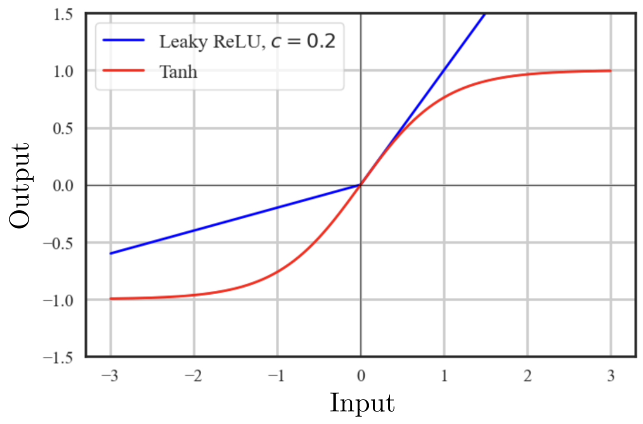
$$y = f\left(\sum_{i=1}^{n} w_i \cdot x_i + b \right), \tag{1.46}$$

whereas x_i are the input values, w_i are the weights, b is the bias term and f is the activation function.

While various activation functions are available, LeakyRELU and Tanh are used in this analysis as shown in figure 1-8 due to their respective advantages. LeakyRELU as activation



A typical neuron as the key building block of ANNs



Activation functions

Figure 1-8: Visualization of neurons of an ANN [3] and exemplary activation functions

function is defined as:

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ cx & \text{if } x \leq 0, \end{cases} \tag{1.47}$$

where c is a constant with $0 < c < 1$. LeakyReLU is used in the hidden layers of both the actor and critic networks. It mitigates the issue of neuron deactivation by allowing a small gradient for negative inputs. Then, tanh as activation function can be written as:

$$f(x) = \tanh(x). \tag{1.48}$$

The tanh function is utilized in the output layer of the actor network, ensuring that the actions are bounded within the desired range.

ANN parameters are usually optimized iteratively using a variant of gradient descent during the training process of the neural network [3].

$$\boldsymbol{\mathcal{W}}^{(l)} \leftarrow \boldsymbol{\mathcal{W}}^{(l)} - \alpha \nabla_{\boldsymbol{\mathcal{W}}^{(l)}} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}), \tag{1.49}$$

$$\boldsymbol{b}^{(l)} \leftarrow \boldsymbol{b}^{(l)} - \alpha \nabla_{\boldsymbol{b}^{(l)}} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}). \tag{1.50}$$

There, α is the step size and $\mathcal{L}(.)$ represents the loss between the ground truth vector $\boldsymbol{y}$ and the estimation vector $\hat{\boldsymbol{y}}$, with l denoting the layer.

Structure of Deep Deterministic Policy Gradient

The structure of DDPG's working principle is visualized in figure 1-9 and it contains the key components of *actor network, critic network, target networks, replay buffer* and *exploration noise.*

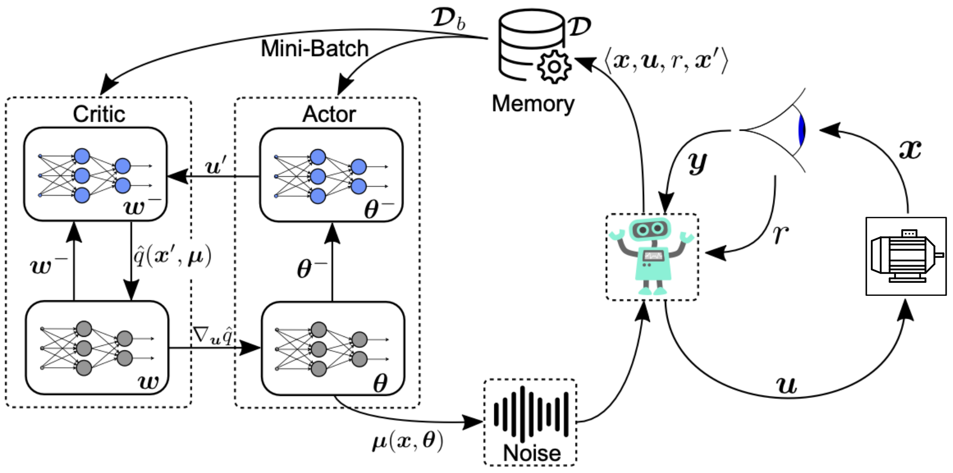


Figure 1-9: Visual summary of DDPG working principle [3]

From the two gradients included in the gradient equation (1.45), the *actor network* $\pi(x, \theta)$ represents the policy, mapping states to actions. It approximates the deterministic policy using an ANN and adjusts the policy parameters in the direction of actions that increase the action-value $q(x, u)$. This process enhances decision performance and provides continuous action outputs for given states as follows:

$$J(\pi_\theta) = \mathbb{E}\left[q(x, \pi(x, \theta))\right]. \tag{1.51}$$

Then, it is updated by maximizing the expected return from the current state as stated in equation (1.45). However, the real action-value function is not initially known, so the *critic network* approximates the action-value function using the ANN, and estimating the quality of actions taken by the policy to guide the policy update as follows:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \left[r + \gamma \max_{u} \hat{q}(\boldsymbol{x'}, u, \boldsymbol{w}) - \hat{q}(\boldsymbol{x}, u, \boldsymbol{w}) \right] \nabla_{\boldsymbol{w}} \hat{q}(\boldsymbol{x}, u, \boldsymbol{w}). \tag{1.52}$$

Then, the critic network is updated by minimizing the loss as follows:

$$\underbrace{\boldsymbol{\mathcal{L}(w)}}_{\text{Generalized Cost Function}} = \Big[\underbrace{\big(r + \gamma q(\boldsymbol{x'}, \boldsymbol{\pi}(\boldsymbol{x'}, \boldsymbol{\theta}), \boldsymbol{w}^{-})\big)}_{\text{Target Network Information}} - \underbrace{q(\boldsymbol{x, u, w})}_{\text{Q-value Estimator}} \Big]^{2}_{\boldsymbol{\mathcal{D}}_b}. \tag{1.53}$$

Furthermore, target networks are introduced to estimate the Q-learning target. The primary purpose of target networks is to stabilize training by reducing large oscillations in the parameters. This is achieved by decoupling the target Q-values from the current Q-values and gradually updating the target network. In the DDPG algorithm, target networks are continuously updated using a low-pass filter characteristic, as described below:

$$\boldsymbol{w}^{-} \leftarrow (1 - \lambda)\boldsymbol{w}^{-} + \lambda \boldsymbol{w}, \tag{1.54a}$$

$$\boldsymbol{\theta}^{-} \leftarrow (1 - \lambda)\boldsymbol{\theta}^{-} + \lambda \boldsymbol{\theta}. \tag{1.54b}$$

Here, $\boldsymbol{w}$ and $\boldsymbol{w}^{-}$ represent the critic weights and delayed critic weights, respectively, while $\boldsymbol{\theta}$ and $\boldsymbol{\theta}^{-}$ represent the policy weights and delayed policy weights, respectively. The parameter λ refers to the equivalent filter constant, a hyper-parameter in the range of $0 < \lambda < 1$. If the λ value is small, the target networks are updated slowly; if it is large, the networks are updated more quickly.

Then, the *replay buffer* $\boldsymbol{\mathcal{D}}$ is a crucial component in DDPG, used to store the transitions $\langle \boldsymbol{x}, \boldsymbol{u}, r, \boldsymbol{x'}, \text{done} \rangle$ observed during training after each step, where 'done' refers to the condition where the environment is either terminated or truncated. This buffer enables off-policy learning by allowing the algorithm to sample and learn from past experiences, rather than relying solely on the most recent transitions. Off-policy learning means that the algorithm can learn from actions generated by a behavior policy, which includes exploration noise, rather than strictly following the current policy being optimized. Ideally, off-policy learning leverages two distinct policies [3] as follows:

- Behavior policy $b(u|x)$: explores the environment to generate diverse experiences.
- Target policy $\pi(u|x)$: learns from those experiences to refine and become the optimal policy.

One of the most challenging dilemmas in RL is balancing between exploration and exploitation. Exploration involves the agent trying various actions to gather information and learn more about the environment, enabling it to discover which actions are likely to yield the highest returns. On the other hand, exploitation involves the agent using the best knowledge gained from previous experience to choose the actions that have provided the highest rewards so far. While this approach can be beneficial for achieving short-term maximum rewards, it may result in missing out on better actions that could yield even higher rewards because the agent avoids trying new actions and sticks to what is already known.

In DDPG, exploration poses a unique challenge due to the deterministic nature of the policy, where the same action is always selected for a specific state. This determinism can limit the algorithm's ability to discover potentially better policies. To address this, an off-policy learning approach is employed. The key idea is to select actions based on a stochastic behavior policy, ensuring sufficient exploration, while simultaneously learning about a deterministic target policy.

Among the available exploration noise types, Gaussian noise distribution is introduced due to its ability to generate temporally correlated noise, which is highly effective for exploring environments in reinforcement learning [21].

$$u_{\text{noisy},k} = u_k + \mathcal{N}(\mu, \sigma^2) \tag{1.55}$$

The equation above (1.55) illustrates how the deterministic policy incorporates noise into the action selection process whereas u_k is the action value at time step k before the noise added, and $u_{\text{noisy},k}$ is the noisy action value at time step k where $\mathcal{N}(\mu, \sigma^2)$, the Gaussian noise sample with mean μ and standard deviation σ is added.

Finally, the training process in DDPG involves iteratively improving the policy by interacting with the environment, collecting experiences, and updating the ANNs that approximate the policy and action-value functions. A more detailed step by step procedure is provided in the pseudo code in Algorithm 1.

Algorithm 1 Deep Deterministic Policy Gradient (DDPG) [3]

```
1: input: diff. deterministic policy function $\boldsymbol{\pi(x,\theta)}$ and action-value function $\hat{q}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w})$
2: parameter: step sizes and filter constant $\{\alpha_w, \alpha_\theta, \lambda\} \in \{\mathbb{R} | 0 < \alpha, \lambda < 1\}$
3: init: weights $\boldsymbol{w} = \boldsymbol{w}^- \in \mathbb{R}^s$ and $\boldsymbol{\theta} = \boldsymbol{\theta}^- \in \mathbb{R}^d$ arbitrarily, memory $\boldsymbol{\mathcal{D}}$
4: for $j = 1, 2, \ldots,$ episodes do
5:     initialize $x_0$;
6:     for $k = 0, 1, \ldots, T-1$ time steps do
7:         $\boldsymbol{u}_k \leftarrow$ apply from $\boldsymbol{\pi}(\boldsymbol{x}_k, \boldsymbol{\theta})$ w/wo noise or from behavior policy;
8:         observe $\boldsymbol{x}_{k+1}$ and $r_{k+1}$;
9:         store tuple $\langle \boldsymbol{x}_k, \boldsymbol{u}_k, r_{k+1}, \boldsymbol{x}_{k+1} \rangle$ in $\boldsymbol{\mathcal{D}}$;
10:        sample mini-batch $\boldsymbol{\mathcal{D}}_b$ from $\boldsymbol{\mathcal{D}}$ (after initial memory warmup);
11:        for $i = 1, \ldots, b$ samples do calculate $Q$-targets
12:            if $\boldsymbol{x}_{i+1}$ is terminal then
13:                $y_i = r_{i+1}$
14:            else
15:                $y_i = r_{i+1} + \gamma \hat{q}(\boldsymbol{x}_{i+1}, \boldsymbol{\pi}(\boldsymbol{x}_{i+1}, \boldsymbol{\theta}^-), \boldsymbol{w}^-)$
16:            end if
17:        end for
18:        fit $\boldsymbol{w}$ on loss $\boldsymbol{\mathcal{L}(w)} = [y_- \hat{q}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})]^2_{\boldsymbol{\mathcal{D}}_b}$ with step size $\alpha_w$;
19:        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_\theta [\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}(\boldsymbol{x}, \boldsymbol{\theta}) \nabla_{\boldsymbol{u}} \hat{q}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})|_{\boldsymbol{u}=\boldsymbol{\pi}(\boldsymbol{x},\boldsymbol{\theta})}]_{\boldsymbol{\mathcal{D}}_b}$;
20:        Update target net. $\boldsymbol{w}^- \leftarrow (1-\lambda)\boldsymbol{w}^- + \lambda \boldsymbol{w}$, $\boldsymbol{\theta}^- \leftarrow (1-\lambda)\boldsymbol{\theta}^- + \lambda \boldsymbol{\theta}$;
21:    end for
22: end for
```

1.4 Basic Setup for training the agent

The DDPG algorithm was implemented using Python 3.9.19, with the setup and training of the DDPG model accomplished through the Stable-Baselines3 library. The AC drive system was simulated using the GEM library. Additionally, several other essential libraries were utilized, as summarized in table 1.4. The algorithm implementation and agent training will be carried out on a workstation with specifications as listed in table 1.5.

Component	**Library/Tool**	**Version**
Programming language	Python	3.9.19
RL library	Stable-Baselines3 [21]	2.3.0
Deep learning framework	PyTorch	2.2.2
Numerical computation	Numpy	1.26.4
Serialization	Cloudpickle	1.6.0
Environment interface	Gymnasium [22]	0.29.1
Legacy environment API	OpenAI Gym	0.17.3
AC drive simulation	gym-electric-motor [12]	2.0.0

Table 1.4: Software components and versions

Workstation Specifications	
CPU	Apple M2
RAM	8 GB
Hard drive memory	256 GB
OS	macOS Sonoma 14.3

Table 1.5: Detail data of workstation

Chapter 2

Reinforcement Learning based Current Control

In this chapter, the CCS current control of the EESM is examined using the DDPG algorithm. The control is performed in rotor-oriented coordinates within this framework. The schematic diagram of the overall control structure is presented in figure 2-1, where a B6 bridge converter is utilized for the stator circuit, and a 4QC converter is applied to the rotor circuit, as discussed in section 1.3.5. The primary objective of this control structure is to accurately track the reference current, ensuring minimal steady-state error and reduced transient time.

2.1 RL-CCS Current Control

As shown in figure 2-1, the input to the control system comprises three reference currents (i^*_{sd}, i^*_{sq}, and i^*_{f}), which serve as the targets for the system to track. To ensure consistency in scale among all states, the reference currents $\boldsymbol{i}^*_{\mathrm{dqf}}$ are normalized. This step is crucial as it prevents features with larger numerical ranges from disproportionately influencing the

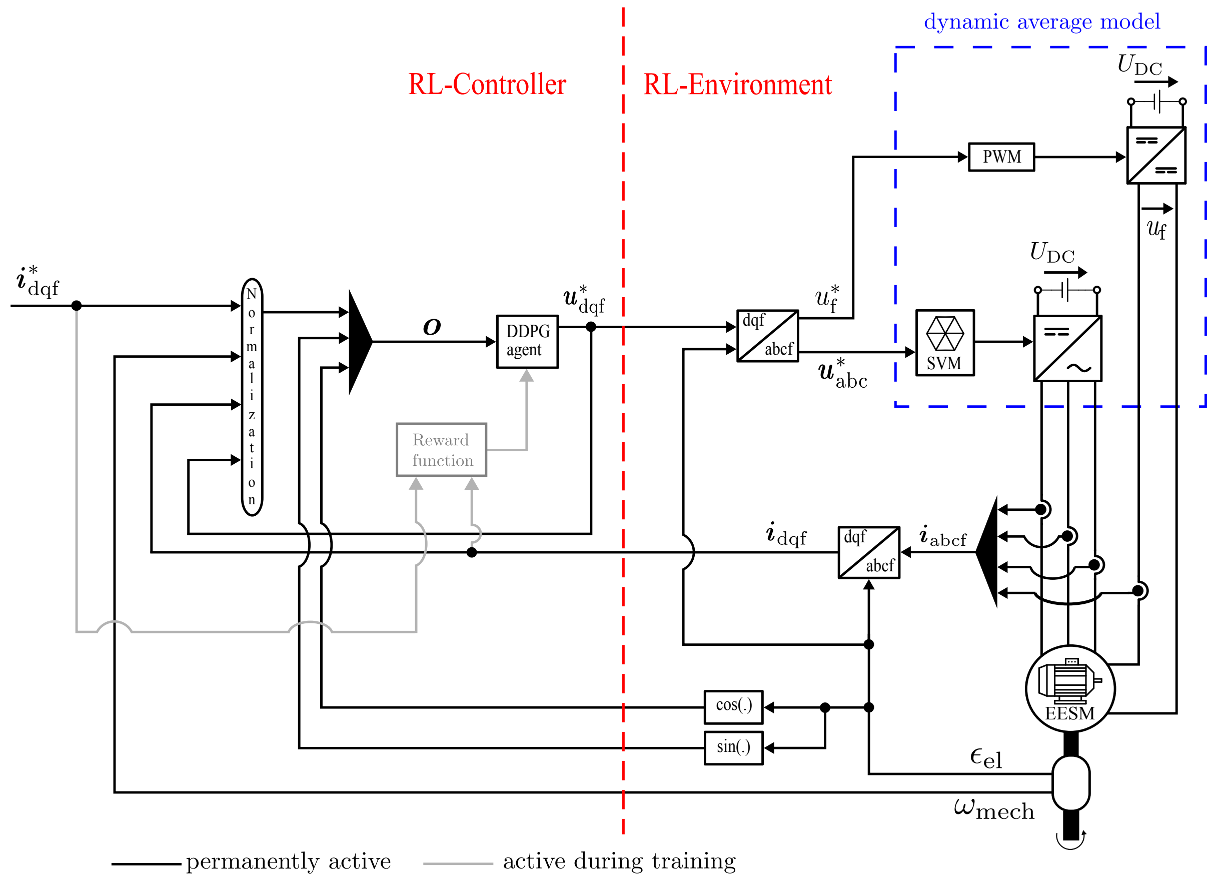


Figure 2-1: Schematic of overall CCS current control structure

training process. The normalization is performed as follows:

$$\tilde{x} = \frac{x}{x_{\max}} \tag{2.1}$$

where $\tilde{x}$ is a general representation of the normalized variables. This normalization is applied not only to the reference inputs but also to the feedback currents ($\boldsymbol{i}_{\mathrm{dqf}}$), the agent-selected actions ($\boldsymbol{u}^*_{\mathrm{dqf}}$), and the rotor speed ($\omega_{\mathrm{mech}}$). The trigonometric functions of the rotor electrical position, $\cos(.)$ and $\sin(.)$, do not require normalization since they naturally fall within the range [-1, 1]. As a result, the normalized ranges for the states are:

$$\tilde{i}^*_{\mathrm{dq}}, \tilde{i}_{\mathrm{dq}}, \tilde{u}^*_{\mathrm{dqf}}, \tilde{\omega}_{\mathrm{mech}}, \cos(.), \sin(.) \in [-1, 1] \;\text{ and }\; \tilde{i}^*_{\mathrm{f}}, \tilde{i}_{\mathrm{f}} \in [0, 1]. \tag{2.2}$$

The system observation vector, denoted as $\boldsymbol{o}$, is then input into the DDPG agent, which has been designed, as discussed in section 1.3.8, to output the optimal control actions $\boldsymbol{u}^*_{\mathrm{dqf}}$. These actions represent the voltage signals in the d-, q-, and f-coordinate frame, which the system applies to achieve the desired current control.

As discussed in the context of finite MDPs, the agent must have access to all relevant system states. Therefore, the observation space is structured as follows:

$$\boldsymbol{o}_k = \left[\tilde{i}_{\mathrm{sd},k} \ \tilde{i}_{\mathrm{sq},k} \ \tilde{i}_{\mathrm{f},k} \ \tilde{\omega}_{\mathrm{mech},k} \ \ \cos\left(\epsilon_{\mathrm{el},k}\right) \ \ \sin\left(\epsilon_{\mathrm{el},k}\right) \ \tilde{i}^*_{\mathrm{sd},k} \ \tilde{i}^*_{\mathrm{sq},k} \ \tilde{i}^*_{\mathrm{f},k} \ \tilde{u}_{\mathrm{sd},k} \ \tilde{u}_{\mathrm{sq},k} \ \tilde{u}_{\mathrm{f},k}\right] \tag{2.3}$$

Since the return determines the agent's performance, the design of the reward function plays a crucial role in optimizing control behavior.

2.2 Reward Design for CCS Current Control

The reward function must be designed to guide the agent in selecting the optimal actions for the control problem under any given circumstance. Since the agent aims to maximize the reward through its interactions with the system, the highest achievable reward should correspond to the optimal operating point. However, caution must be exercised in the design of the reward function to include physical boundaries, ensuring the safe operation of the machine within its available range. As mentioned earlier, the agent will apply the selected actions $\boldsymbol{u}_{\mathrm{dqf}}$, and any violation of the predefined constraints will result in a penalty or negative reward.

1: Excess rotor current region

In an EESM drive, the rotor circuit is supplied by DC power, and it is crucial to keep the excitation current below its upper limit to prevent over-current operation. This constraint is incorporated into the reward function, where any excess excitation current results in a negative reward proportional to the amount by which the current exceeds the limit:

$$\text{if} \quad (i_{\mathrm{f},k} > i_{\mathrm{f,lim}}) \Rightarrow \quad r_k = \left(-1 - \frac{i_{\mathrm{f},k}}{i_{\mathrm{f,lim}}}\right). \tag{2.4}$$

2: Excess stator current region

Similarly, the stator current of the EESM must remain within the allowable current circle. This is another constraint factored into the reward design for current control. Operating with an over-current condition in the stator circuit will result in a negative reward, scaled by how far the current exceeds the limit:

$$\text{if} \quad (i_{\mathrm{s},k} > i_{\mathrm{s,lim}}) \Rightarrow \quad r_k = \left(-0.1 - \frac{i_{\mathrm{s},k}}{i_{\mathrm{s,lim}}}\right). \tag{2.5}$$

Whenever the constraints in the equations (2.4) and (2.5) are violated, the system will be terminated, and the corresponding penalty will be applied.

3: Unfavourable rotor current region

As discussed in section 1.3.5, the rotor current should also be maintained above zero, with currents below zero considered undesirable. To discourage operation in this unfavorable region, a negative reward is assigned proportional to the magnitude of the current below zero:

$$\text{if} \quad (i_{\mathrm{f},k} < 0) \Rightarrow \quad r_k = -0.1 + 2\left(\frac{i_{\mathrm{f},k}}{i_{\mathrm{f,lim}}}\right). \tag{2.6}$$

4: Tracking the references

After ensuring safe operating conditions through penalties for constraint violations, the reward function must guide the agent to achieve its primary objectives: accurate stator and rotor current tracking. To accomplish this, the agent is rewarded based on how closely the currents (i_{sd}, i_{sq}, i_{f}) follow its references (i^*_{sd}, i^*_{sq}, i^*_{f}). The reward is defined as follows:

$$\text{else} \quad \Rightarrow \quad r_k = (1-\gamma)\,(1-e_{\mathrm{sum}})\,, \tag{2.7}$$

$$\Rightarrow \quad r_k \in \Bigg[\min\left[(1-e_{\mathrm{sum}})\,(1-\gamma)\right], \quad 1-\gamma\Bigg]. \tag{2.8}$$

In this context, e_{sum} represents the combined error between the actual currents and reference currents, calculated as:

$$e_{\mathrm{sum}} = e_{\mathrm{sd}} + e_{\mathrm{sq}} + 2e_{\mathrm{f}} \in (0,8)\,, \tag{2.9}$$

where,

$$e_{\mathrm{sd}} = |\tilde{i}^*_{\mathrm{sd}} - \tilde{i}_{\mathrm{sd}}| \in (0,2)\,, \tag{2.10}$$

$$e_{\mathrm{sq}} = |\tilde{i}^*_{\mathrm{sq}} - \tilde{i}_{\mathrm{sq}}| \in (0,2)\,, \tag{2.11}$$

$$e_{\mathrm{f}} = |\tilde{i}^*_{\mathrm{f}} - \tilde{i}_{\mathrm{f}}| \in (0,2)\,. \tag{2.12}$$

Here, e_{sd}, e_{sq} and e_{f} are the errors between the reference and actual currents of their respective axes. It is important to note that e_{f} is multiplied by 2 during the error summation to place more emphasis on the f-axis current error, recognizing that excitation current is generally more challenging to control than stator current (details discussed in section 2.3).

The reward design outlined in steps 1 through 4 systematically ensures that the RL agent not only operates the EESM within safe boundaries but also optimizes its control performance by balancing both safety and precision in a structured manner.

2.3 Training the DDPG Agent

To balance computational time and agent performance, the agent in this analysis is trained for $90 \cdot 10^4$ environment steps. Given a sampling frequency of 10 kHz, the physical training duration would correspond to 75 seconds. However, since the training is conducted in an asynchronous simulation rather than a real-time environment, simulating these 75 seconds takes approximately 137 minutes on the workstation. The selected hyper-parameters and the ANN architecture to run the DDPG algorithm are shown below.

Overview of Selected Hyper-parameters Set

Hyper-parameters are settings or configurations that are determined before the learning process begins and play a crucial role in shaping the behavior of the algorithm. These parameters, which are integral to the equations presented in section 1.3.8, need to be carefully tuned to optimize the performance of the DDPG algorithm. The selected hyper-parameters for this CCS current control are summarized in the following table 2.1.

Symbol	**Description**	**Selected set**
γ	discount factor	0.9
$\mathcal{D}_b$	mini-batch size	256
$\mathcal{D}$	replay buffer size	$90 \cdot 10^4$
K	maximum episode length	$5 \cdot 10^3$
M	total training steps	$90 \cdot 10^4$
α_{actor}	actor learning rate	$1 \cdot 10^{-5}$
α_{critic}	critic learning rate	$1 \cdot 10^{-4}$
$F_{\text{prediction}}$	prediction networks update parameter	1 time step
N_{train}	memory warm-up	$5 \cdot 10^3$
ν	exploration noise	see figure 2-2
f_{s}	sampling frequency	10 kHz

Table 2.1: Selected hyper-parameters set for CCS current control

Regarding exploration noise, Gaussian noise, as discussed in section 1.3.8, is utilized with zero mean values for all three actions (u_{sd}, u_{sq}, u_{f}). The scale of this noise, represented by the standard deviation σ, varies over the course of training steps, as illustrated in figure 2-2. The evolution of σ can be categorized into three distinct stages:

1. Initial Stage: A low deviation value of $\sigma = 0.3$ is employed during the early training steps. This allows the agent to familiarize itself with the environment and supports stable learning.

2. Exploration Stage: As training progresses, σ increases linearly to its maximum value, enabling full exploration of the action space.

3. Exploitation Stage: In the later training steps, σ decreases exponentially, approaching zero. This reduction in noise facilitates exploitation, allowing the agent to fine-tune its policy based on learned experiences.

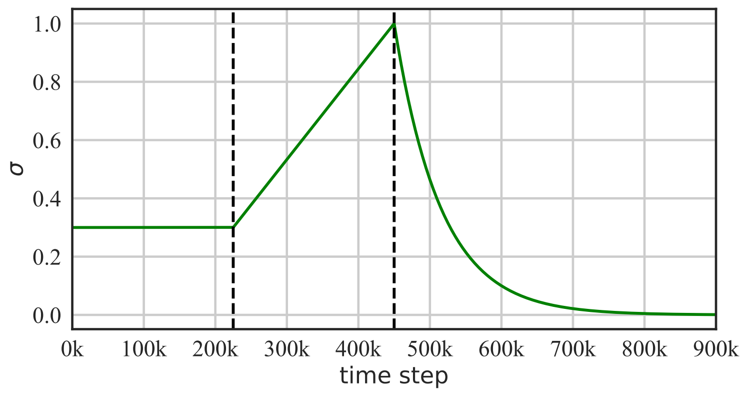


Figure 2-2: Variation of the standard deviation of the Gaussian distribution over time steps

By dynamically adjusting σ throughout these stages, the agent will be able to balance exploration and exploitation.

Artificial Neural Network Architecture

Moreover, as discussed in section 1.3.8, the ANN plays an important role in the DDPG algorithm by enabling function approximation, which constitute the actor-critic architecture. However, it is essential to properly balance the network's size: if the network is too small, it may not be sufficient to handle the complexity of the environment; conversely, if it is unnecessarily large, it could require excessive training time. Therefore, the ANNs for this current control are designed as illustrated in table 2.2.

Network	**Layer Type**	**Neurons**	**Activation Function**	**Definition**
Actor	input layer	12	-	-
	hidden layer 1	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 2	128	LeakyReLU	$\max(0.2x, x)$
	output layer	3	Tanh	$\tanh(x)$
Critic	input layer	15	-	-
	hidden layer 1	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 2	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 3	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 4	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 5	128	LeakyReLU	$\max(0.2x, x)$
	output layer	1	Linear	x

Table 2.2: Artificial neural network configuration for CCS current control

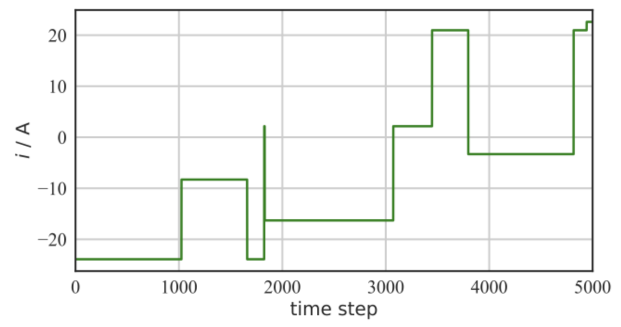
With the selection of hyper-parameters complete and the ANN architecture established, the DDPG algorithm is now prepared to interact with the environment and begin the training process. The EESM parameters, as detailed in the accompanying table 1.2, will be utilized as the environment for this CCS current control task.

Patterns of References

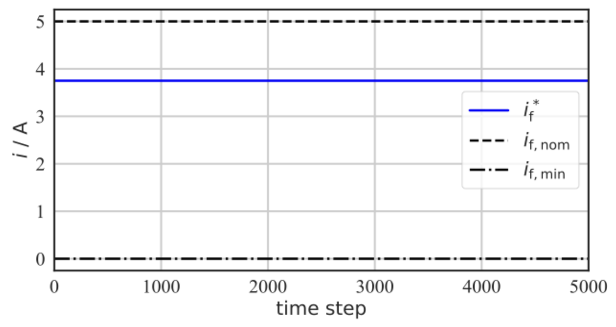
As illustrated in figure 2-1, the reference currents (i^*_{sd}, i^*_{sq}, i^*_{f}) serve as inputs to the control structure. The behavior of these reference currents during an episode of the training process is shown in the figure 2-3.

The time constants for the stator and rotor are critical in determining the behavior of the reference currents, as represented by the following equations:

$$\tau_{\mathrm{sd}} = \frac{L_{\mathrm{d}}}{R_{\mathrm{s}}\sigma_{\mathrm{l}}} = 114 \text{ ms} \tag{2.13}$$



Reference d- and q-axes current sample



Reference excitation current sample

Figure 2-3: Reference inputs for an exemplary episode in the CCS current control structure

$$\tau_{\mathrm{sq}} = \frac{L_{\mathrm{q}}}{R_{\mathrm{s}}} = 9.84 \text{ ms} \tag{2.14}$$

$$\tau_{\mathrm{f}} = \frac{L_{\mathrm{f}}}{R_{\mathrm{f}}\sigma_{\mathrm{l}}} = 206 \text{ ms} \tag{2.15}$$

In these equations, τ_{sd}, τ_{sq} and τ_{f} represent the time constants of the stator and rotor circuits along the d-, q-, and f-axes, respectively. Then σ_{l} is the coupling or leakage factor between d- and f-axes which is defined as:

$$\sigma_{\mathrm{l}} = 1 - \frac{3L_{\mathrm{m}}^2}{2L_{\mathrm{d}}L_{\mathrm{f}}}. \tag{2.16}$$

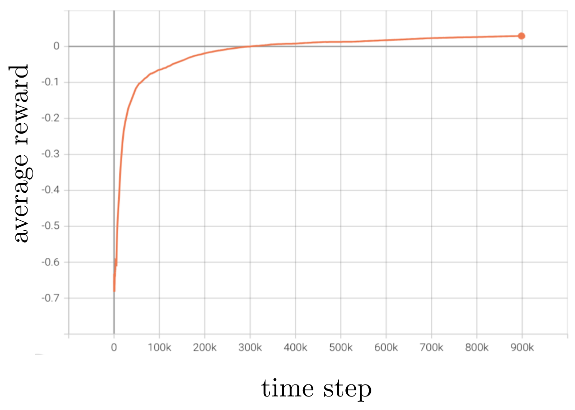
Notably, the rotor time constant τ_{f} is longer than the stator time constants τ_{sd} and τ_{sq} indicating a longer transient period for the rotor to reach a stable state. Given this difference in time constants, it was chosen to ensure i_{f}^* remains unchanged until i_{f} has reached steady state. To ensure the agent experiences the steady-state response of the excitation current, the episode length is set to $5 \cdot 10^3$ time steps as shown in table 2.1, which is approximately twice the duration of the excitation current transient in simulation time steps. This approach ensures that the agent can accurately learn and understand the system dynamics.

To accommodate these dynamics, the reference currents i_{sd}^* and i_{sq}^* are introduced as step-wise inputs as shown in figure 2-3, with amplitudes varying between positive and negative nominal values. The time between these reference value steps changes randomly. Meanwhile,

i_f^* is held constant throughout each episode, although its value randomly shifts between zero and the nominal level across different episodes. This setup offers a controlled but diverse training environment, allowing the agent to learn the system dynamics effectively.

Average Reward and Episode Length during Training

The behavior of the average reward and episode length throughout the training process is illustrated in figure 2-4. The average reward refers to the mean episodic reward, while the average episode length represents the mean number of steps taken per episode before termination or completion [21]. Both metrics are averaged over the most recent 100 episodes by the Stable-Baselines3 library. Initially, when fewer than 100 episodes are available, the averages are calculated based on the available episodes. Once more than 100 episodes are completed, the averages are continuously updated based on the last 100 episodes. This approach is useful because it helps track the agent's recent learning performance rather than averaging out performance over the entire training.



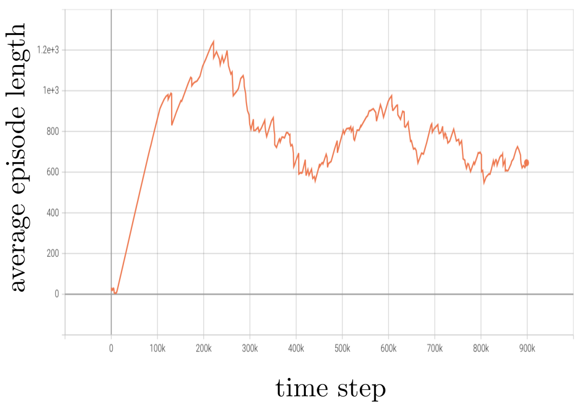


Figure 2-4: Average reward and episode length over the training steps

As observed, the reward curve exhibits a gradual improvement over time, indicating that the agent is progressively learning to interact more effectively with the environment. Meanwhile, the mean episode length fluctuates during the exploration stage, reflecting the agent's ongoing efforts to balance exploration and exploitation. While the trends suggest that further training could potentially lead to continued improvements, the current results indicate that the agent has reached a level of competence that warrants validation in an environment with different references. The subsequent section will assess whether the training duration was sufficient for the agent to fully grasp the system's dynamics and perform reliably in different conditions.

2.4 Validation and Performance Analysis

To assess the agent's adaptability after training for $90 \cdot 10^4$ time steps, the environment was changed to a validation setting with new reference input currents, as shown in figure 2-5.

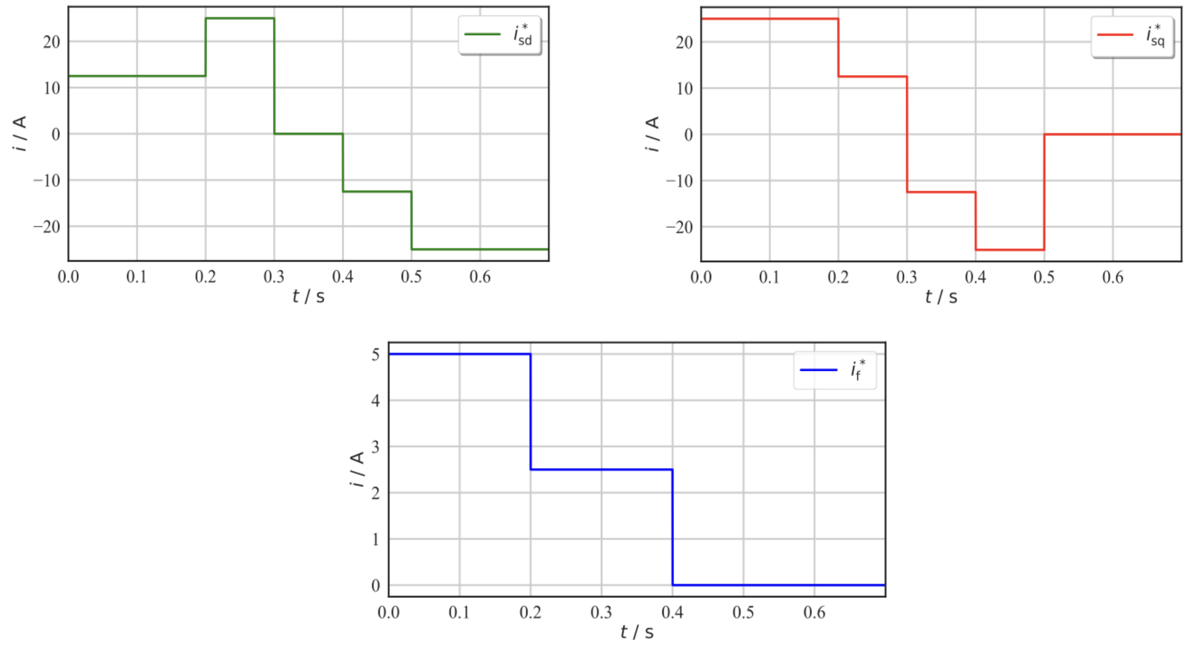


Figure 2-5: Reference inputs to a validation environment

In figure 2-5, the reference currents i^*_{sd} and i^*_{sq} cover the entire range from positive to negative nominal values, while i^*_{f} varies from nominal to zero. Based on these references, two validation cases were conducted: one at a negative low speed and the other at a positive high speed, to evaluate the agent's control capability across a wide speed range. The validation case at negative low speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$ will be discussed in detail below, while the case at positive high speed is presented in the Appendix A.

Analysis of the results

The validation results are presented in figure 2-6, which includes the actions (u_{sd}, u_{sq}, u_{f}) and their corresponding current responses. At first glance, it is clear that the agent successfully tracks all three reference currents with low error, while maintaining the stator and rotor voltages within the defined ranges of $u_{\mathrm{dq}} \in [-100, 100]$ and $u_{\mathrm{f}} \in [-200, 200]$.

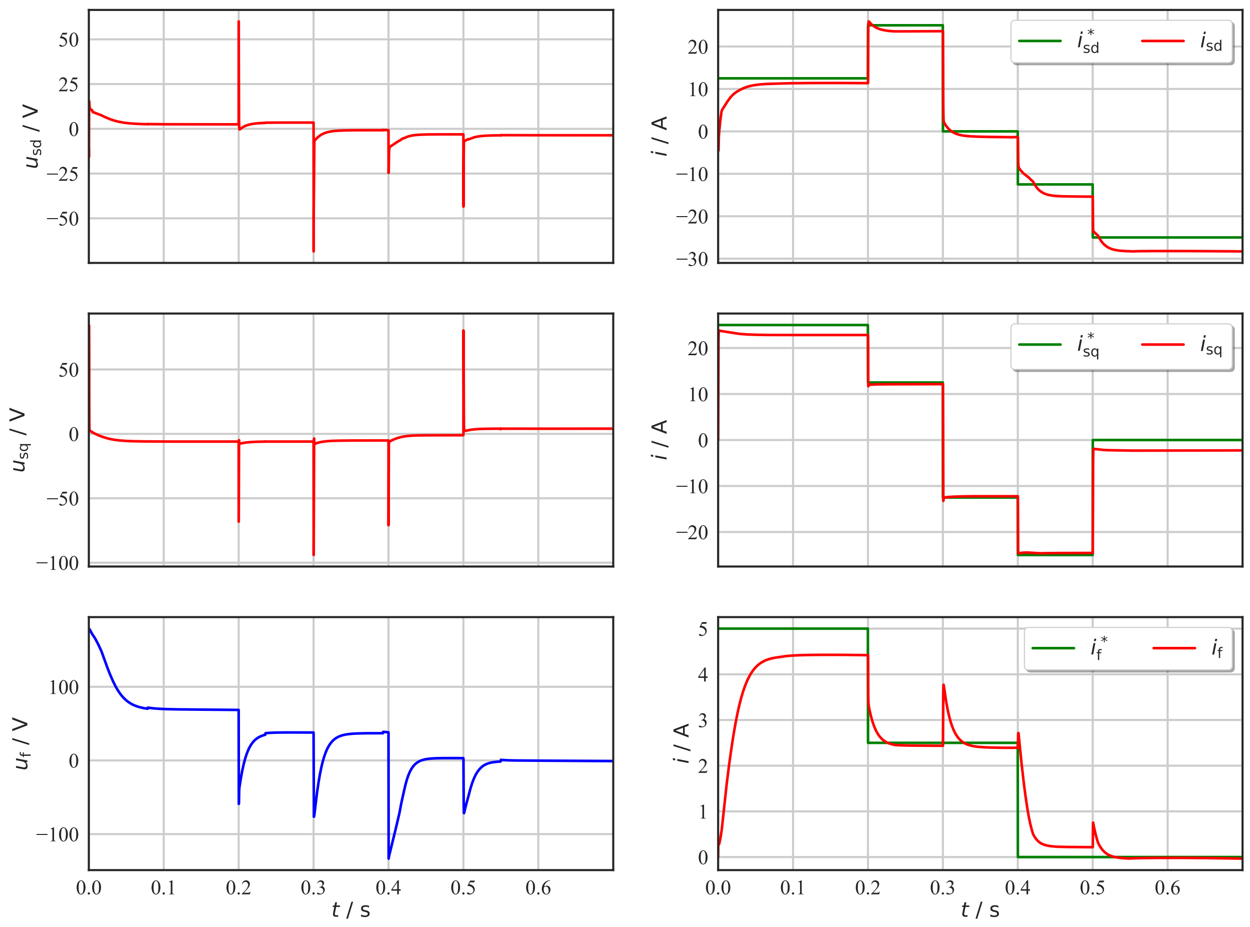


Figure 2-6: Validation results of the agent at a speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$

As mentioned in equation (2.15), the longer rotor time constant significantly impacts the behavior of i_{f} as it tracks its reference i^*_{f}. During the initial simulation period (less than 0.2 seconds), while i_{sd} and i_{sq} quickly reach steady state in tracking their respective references, i_{f} remains in a transient state due to the longer rotor time constant. This difficulty in controlling the excitation current is why more weight was assigned to it during the reward design.

Moreover, the coupling effect between the d- and f-axes, as discussed in the EESM dynamics, is clearly observed. At the simulation time of 0.3 seconds, i_{sd} drops from its nominal value to zero, which causes a spike in the f-axis current. In response, the agent applies a negative u_{f} to push i_{f} back to its steady-state value, with the excitation current gradually recovering.

Apart from the challenges posed by the longer rotor time constant and the coupling effect, there is little else to note. The agent demonstrates that it has sufficiently learned the system's dynamics, reacting appropriately to changes and successfully achieving its primary control objectives.

2.5 Key Takeaways of the Chapter

This chapter marks the foundational step toward achieving optimal torque control of an EESM using RL. The analysis conducted demonstrated that the DDPG agent is capable of navigating the complex and multi-dimensional space of the EESM. It effectively controls the d-, q-, and f-axes currents by applying the corresponding actions, addressing the challenges posed by the strong coupling effects between axes and the longer rotor time constant.

This analysis is important for optimal torque control because, in real-world applications, torque sensors are often not utilized, and current sensors are the primary means of feedback. As a result, the only way to estimate the produced electromagnetic torque is by calculating it from i_{sd}, i_{sq} and i_{f} using the torque production equation (1.26). Therefore, it is vital for the agent to efficiently learn the system's dynamics and effectively control the currents as a foundational step before transitioning to torque control.

Building on this foundational understanding, the next chapter will explore torque control using a PI-assisted RL controller. In this scenario, the agent will infer its torque tracking performance indirectly from current measurements, as no torque sensor will be used, and the agent will not receive direct torque feedback. A torque sensor is available only during the training phase, so that the reward can be calculated accordingly. This setup will test the agent's ability to control torque based solely on current feedback.

Chapter 3

Torque Control with Idealized Excitation Circuit Current Control

Building on the success demonstrated in the previous chapter, where the DDPG agent effectively navigated the complex environment of the EESM and accurately tracked the i_{sd}^{*}, i_{sq}^{*} and i_{f}^{*} in a continuous action space, this chapter shifts the focus from CCS current control to CCS torque control. Unlike current control, where the agent applies voltage to track the measured current, torque control presents a more complex challenge, as the agent does not have knowledge of the torque directly. However, torque knowledge will be used for the reward generation. To facilitate this transition smoothly, the excitation circuit will first be controlled to maintain a constant field current using a proportional-integral (PI) controller. This approach simplifies the problem by fixing the excitation current, allowing the agent to focus on manipulating the d- and q-axes currents to track the reference torque, similar to the control strategy used in PMSMs [8]. The overall control structure is illustrated in figure 3-1.

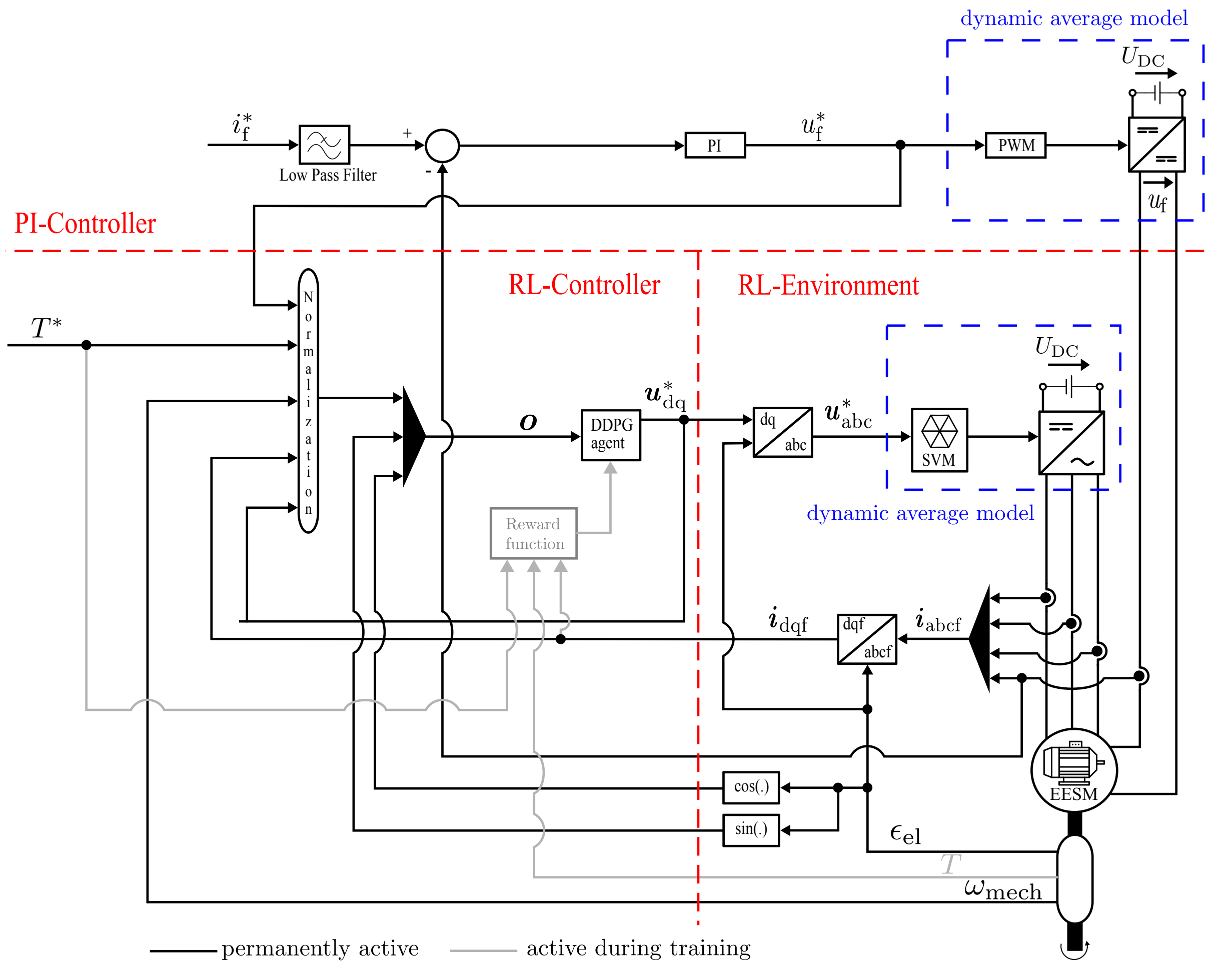


Figure 3-1: Schematic of the overall CCS torque control structure with PI controller

3.1 RL-CCS Torque Control

As illustrated in figure 3-1, the control structure is composed of three main parts. The conventional PI controller is employed to maintain a constant excitation current across all operating points, regardless of changes in the reference torque. While this approach may not optimize the rotor circuit in terms of efficiency, for the DDPG agent, the environment will behave similarly to a PMSM, where only the d- and q-axes currents vary in response to changes in the reference torque.

In the PI controller section, a low-pass filter is employed to ensure that changes in the reference excitation current occur gradually, following an exponential curve rather than a step input. This approach reduces the likelihood of overshoot and oscillation in the tracking current, while also mitigating the disturbance impact on the d- and q-axes currents, which are common issues when the reference is introduced as a step input. The tuning process for this PI controller is further discussed in section 3.2.1.

In the RL-controller section, the input is the reference torque, which, as usual, must be normalized as discussed in section 2.1. Although the torque production of the EESM involves three currents, as described in equation (1.26), the DDPG agent only provides u_{sd} and u_{sq} to achieve the required currents in the d- and q-axes, while the f-axis current is regulated by the PI controller. It is important to note that for the agent to learn the system's dynamics properly, it must have knowledge of the applied rotor voltage and current, even though the field current is controlled by the PI controller. Additionally, since a torque sensor would not be employed in a practical application after the commissioning of the RL controller, feedback torque is only utilized during training for the reward function, meaning the agent does not observe the produced electromagnetic torque. Consequently, torque is not included in the observation vector, which is defined as follows:

$$\boldsymbol{o}_k = \left[\tilde{i}_{\mathrm{sd},k} \;\; \tilde{i}_{\mathrm{sq},k} \;\; \tilde{i}_{\mathrm{f},k} \;\; \tilde{\omega}_{\mathrm{mech},k} \;\; \cos\left(\epsilon_{\mathrm{el},k}\right) \;\; \sin\left(\epsilon_{\mathrm{el},k}\right) \;\; \tilde{T}^*_k \;\; \tilde{u}_{\mathrm{sd},k} \;\; \tilde{u}_{\mathrm{sq},k} \;\; \tilde{u}_{\mathrm{f},k}\right]. \tag{3.1}$$

$$\tilde{T}, \tilde{T}^* \in [-1, 1] \tag{3.2}$$

The torque is normalized to a range between -1 and 1, and the other states are consistent with the approach mentioned previously 2.1.

3.2 Controller Design Approach

In the CCS torque control structure, both the PI controller and the RL controller, as depicted in figure 3-1, play crucial roles. This section will discuss the tuning method for the PI controller and the reward design strategy for the DDPG agent.

3.2.1 Tuning the PI Controller

Due to the strong coupling between the d- and f-axes as stated in equations (1.23), any change in the d-axis voltage can cause fluctuations in the field current. While implementing a decoupling network could mitigate this issue, it would introduce additional complexity to the PI controller, which is not preferred for this analysis. Moreover, allowing the coupling effects to persist enables the agent to learn the impact of field current fluctuations on the d-axis current, which is valuable for understanding the system's dynamics. To counteract these fluctuations, the PI-controller is tuned with very high bandwidth, allowing the current to recover quickly to its steady state after a disturbance. This PI controller can be easily tuned by treating the rotor circuit of the EESM as a plant, as illustrated in the closed-loop control structure shown in figure 3-2.

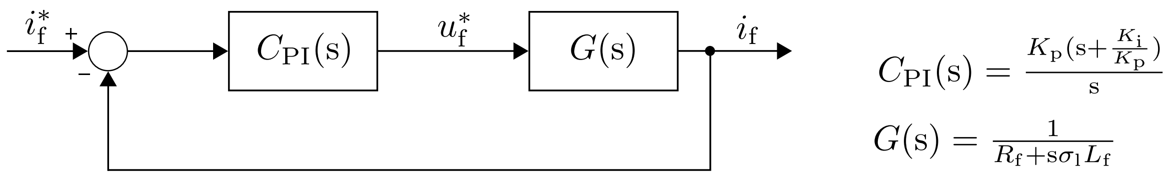

i_f^*
+
−
$C_\mathrm{PI}(\mathrm{s})$
u_f^*
$G(\mathrm{s})$
i_f
$C_\mathrm{PI}(\mathrm{s}) = \frac{K_\mathrm{p}(\mathrm{s}+\frac{K_\mathrm{i}}{K_\mathrm{p}})}{\mathrm{s}}$
$G(\mathrm{s}) = \frac{1}{R_\mathrm{f}+\mathrm{s}\sigma_\mathrm{l}L_\mathrm{f}}$

Figure 3-2: Schematic of PI controller with plant

As depicted, the controller has one pole at the origin and one zero located at $-\frac{K_\mathrm{i}}{K_\mathrm{p}}$, while the system pole is positioned at $-\frac{R_\mathrm{f}}{\sigma_\mathrm{l}L_\mathrm{f}}$. A second-order system with a natural frequency equation is used to tune the controller, as it provides a degree of freedom through the damping factor. Therefore, the denominator of the closed loop transfer function from figure 3-2 is used to compare with the second order equation to calculate the values of K_p and K_i gains as follow:

$$\mathrm{denom}\left(\frac{C_\mathrm{PI}(s)\,G(s)}{1+C_\mathrm{PI}(s)\,G(s)}\right) = \mathrm{s}^2 + 2\zeta\omega_\mathrm{n}\mathrm{s} + \omega_\mathrm{n}^2, \tag{3.3}$$

$$K_\mathrm{p} = 2\zeta\omega_\mathrm{n}\sigma_\mathrm{l}L_\mathrm{f} - R_\mathrm{f}, \tag{3.4}$$

$$K_\mathrm{i} = \omega_\mathrm{n}^2\sigma_\mathrm{l}L_\mathrm{f}. \tag{3.5}$$

The parameters used for tuning the PI controller are summarized in table 3.1.

Symbol	**Description**	**Value**	**Unit**
f	working bandwidth	500	Hz
ω_n	natural frequency	3141.59	s^{-1}
ζ	damping factor	1	-
K_p	proportional gain	$1.46 \cdot 10^3$	VA^{-1}
K_i	integral gain	$2.32 \cdot 10^6$	$\mathrm{V}(\mathrm{A} \cdot \mathrm{s})^{-1}$

Table 3.1: Parameters tuning PI

As mentioned, the PI controller is designed to be as simple as possible to minimize complexity. A damping factor of 1 is chosen to eliminate overshoot, with the trade-off of a potentially slower response [23]. However, the use of a high working bandwidth compensates for this, allowing the system to quickly recover from any transients in the d-axis current. This rapid recovery results in significantly high voltage levels in excitation circuit, therefore the voltage limit is removed in rotor circuit to accommodate the required dynamic adjustments. Otherwise, it would result transients in stator circuit.

3.2.2 Reward Design for PI-assisted RL Control

Since there is no predefined knowledge about the torque control loop, the reward function will play a crucial role in guiding the agent to understand the system's dynamics. Therefore, it is essential to consider all relevant factors in the design of the reward.

As discussed in the previous chapter, the reward design must include physical limitations to ensure safety, with penalties applied for any violation of these constraints. In this torque control scenario, efficiency is also a critical factor, meaning that behaviors that decrease efficiency will be discouraged. Furthermore, since the EESM will be treated similarly to a PMSM in this analysis, the reward design approach used in [8] will be adopted. The reward structure is illustrated in figure 3-3.

In the PMSM reward design [8], the primary operational limit is set by the available

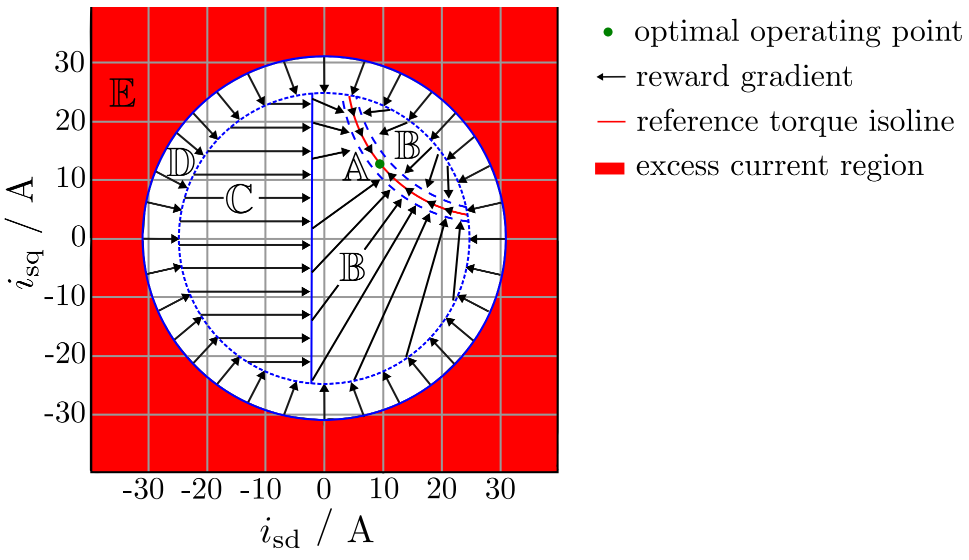


Figure 3-3: Schematic of the reward design approach

stator current, which must not be exceeded to prevent overheating and potential damage to the drives and power electronic converters (region 𝔼). Then, the nominal current must remain below the maximum limit to ensure safe and long-term operation. Although it is allowed to go over the nominal current for a short time, it is not preferable to operate in that region for a long run (region 𝔻). Once the current is within the nominal range, the d- and q-axes currents should be adjusted to ensure that the motor operates with the stator current positioned on the right side of the dq-plane.

While negative i_{sd} is preferred in PMSM operation, positive i_{sd} is more effective for EESM torque production, as indicated by the torque equation (1.26). It is important to note that this positive i_{sd} is specifically chosen because only operation in the linear region is focused in this thesis. Therefore, operating with negative d-axis current will be restricted (region ℂ). However, to provide the agent with some flexibility in tracking low reference torques smoothly, slight deviations below zero will be allowed. The primary objective of this control strategy is to track the reference torque as closely as possible (region 𝔹) while minimizing the stator current, which aligns with the maximum torque per stator current operation (region 𝔸).

The operating regions will be prioritized accordingly, with region $\mathbb{A}$ being the least critical and region $\mathbb{E}$ being the most crucial for safety concerns. This prioritization will be reflected in the reward design, ensuring that the agent learns to operate the EESM safely and efficiently while achieving optimal torque tracking.

1: Excess current region, $\mathbb{E}$

If the stator current enters this region (violating the constraints), it will cause the system to shut down, corresponding to the termination of the episode in the RL environment. The agent must temporarily stop learning and start a new episode. Since this scenario is not desirable in the future, a penalty reward will be applied.

$$\text{if} \quad (i_{\text{s},k} > i_{\text{lim}}) \Rightarrow \quad r_k = -1 \tag{3.6}$$

2: Short-time over-current region, $\mathbb{D}$

To prevent operation in the over-current region, the reward will decrease as the stator current exceeds the nominal limit, emphasizing the importance of safety. This design guides the agent to avoid running the machine in an over-current region, thereby preventing system overload.

$$\text{if} \quad (i_{\text{nom}} < i_{\text{s},k} < i_{\text{lim}}) \Rightarrow \quad r_k = \left(1 - \frac{i_{\text{s},k} - i_{\text{nom}}}{i_{\text{lim}} - i_{\text{nom}}}\right) \frac{1-\gamma}{2} - (1-\gamma) \tag{3.7}$$

$$\Rightarrow \quad r_k \in \left[-(1-\gamma), \quad -\frac{1-\gamma}{2}\right] \tag{3.8}$$

3: Unfavorable efficiency region, $\mathbb{C}$

After safety constraints are set, it is important to eliminate the behaviours which will reduce the efficiency. It is possible to operate with both positive and negative d-axis current safely, but the agent should operate the machine in the right side of dq plane for the intended operation region with respect to the selected EESM geometry.

$$\text{if} \quad (i_{\text{s},k} < i_{\text{nom}}) \quad \text{and} \quad (i_{\text{sd},k} < 0) \Rightarrow \quad r_k = \left(1 + \frac{i_{\text{sd},k}}{i_{\text{lim}}}\right)\frac{1-\gamma}{2} - \frac{1-\gamma}{2} \tag{3.9}$$

$$\Rightarrow \quad r_k \in \left[-\frac{1-\gamma}{2}, \quad 0\right] \tag{3.10}$$

4: Desired operating region, $\mathbb{B}$

In this region, the reward increases as the error decreases, meaning the agent will receive a higher reward for achieving lower torque tracking error (better performance, higher reward).

$$\text{if} \quad (i_{\text{s},k} < i_{\text{nom}}) \quad \text{and} \quad (i_{\text{sd},k} > 0) \quad \text{and} \quad (|T_k^* - T_k| > T_{\text{tol}}):$$

$$\Rightarrow \quad r_k = \left(1 - \left|\frac{T_k^* - T_k}{2T_{\text{lim}}}\right|\right)\frac{1-\gamma}{2} \tag{3.11}$$

$$\Rightarrow \quad r_k \in \left[0, \quad \frac{1-\gamma}{2}\right] \tag{3.12}$$

where T_{tol} = torque control tolerance

5: Reference torque isoline, $\mathbb{A}$

To promote efficiency, the reward will increase as the stator current decreases, following the principles of the MTPC method with stator current. This approach encourages the agent to minimize stator current usage while still accurately tracking the reference torque.

$$\text{if} \quad (i_{\text{s},k} < i_{\text{nom}}) \quad \text{and} \quad (i_{\text{sd},k} > 0) \quad \text{and} \quad (|T_k^* - T_k| < T_{\text{tol}}):$$

$$\Rightarrow \quad r_k = \left(1 - \frac{i_{\text{s},k}}{i_{\text{lim}}}\right)\frac{1-\gamma}{2} + \frac{1-\gamma}{2} \tag{3.13}$$

$$\Rightarrow \quad r_k \in \left[\frac{1-\gamma}{2}, \quad 1-\gamma\right] \tag{3.14}$$

Although this thesis focuses on operation under base speed, the reward design from 1 to 5 is, in theory, versatile enough to effectively cover all operating points of the EESM, including both constant torque and constant power regions. Importantly, this reward structure does not rely on specific machine parameters, enabling the development of a data-driven controller that can operate efficiently with only the stator current limit and nominal values for normalization.

3.3 Training the DDPG Agent

Generally, the longer the agent is trained, the better its performance should be. However, extended training steps require more computational time and a workstation with high computational capabilities. Given this trade-off, the agent in this study is trained for $75 \cdot 10^4$ time steps. With a sampling frequency of 10 kHz, this training duration would equate to 75 seconds. However, the training took 150 minutes to simulate 75 seconds because an asynchronous simulation was used instead of a real-time simulation environment. The hyper-parameters and ANN architecture for implementing the DDPG algorithm have been appropriately configured.

Overview of Selected Hyper-parameters Set

With the shift in control objective from current to torque, a new set of hyper-parameters specifically suited to the torque control structure is selected. This change is necessary because the dynamics and control goals of torque regulation differ significantly from current control, requiring adjustments to optimize the learning process. The selected hyper-parameters, based on error and trail, for the RL integrated with the PI controller for CCS torque control are presented in table 3.2.

Symbol	**Description**	**Selected set**
γ	discount factor	0.9
$\mathcal{D}_b$	mini-batch size	64
$\mathcal{D}$	replay buffer size	$75 \cdot 10^4$
K	maximum episode length	$6 \cdot 10^3$
M	total training steps	$75 \cdot 10^4$
α_{actor}	actor learning rate	$1 \cdot 10^{-4}$
α_{critic}	critic learning rate	$1 \cdot 10^{-3}$
$F_{\text{prediction}}$	prediction networks update parameter	1 time step
N_{train}	memory warm-up	$6 \cdot 10^3$
ν	action noise	see figure 2-2
T_{tol}	torque control tolerance	0.1%
f_{s}	sampling frequency	10 kHz

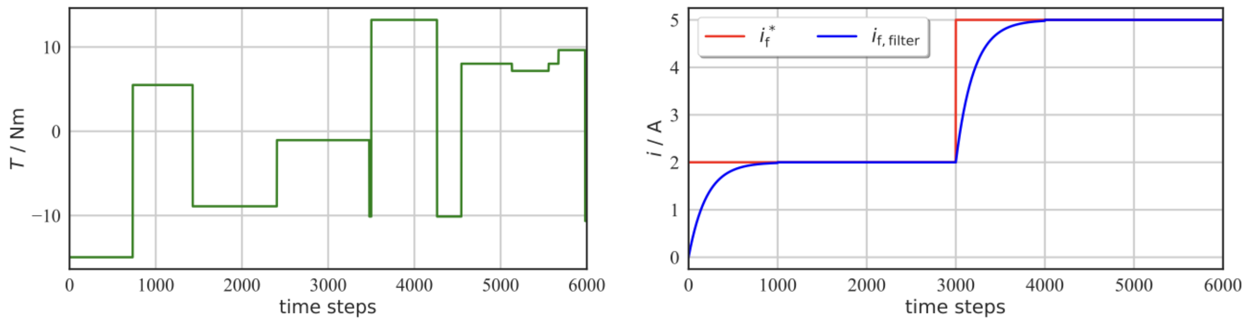
Table 3.2: Selected hyper-parameters set for PI-assisted RL control

Artificial Neural Network Architecture

Although torque control imposes greater computational demands compared to current control, the agent in this analysis is only required to control the d- and q-axes currents. Given this, the previously utilized ANNs from the current control phase are expected to be sufficient. Therefore, the same ANNs are employed here without modification.

Patterns of References

Since the PI controller is used for the excitation circuit, the reference field current is set manually, as it is not part of the RL controller, while the reference torque is provided as an input to the RL controller. A sample of both reference inputs is shown in figure 3-4. To ensure the agent experiences all possible torque reference inputs during training and understands different operating points, the reference torque is introduced as a step-wise input. Additionally, it is crucial for the agent to recognize that excitation current might vary in real-world applications; hence, the reference current is adjusted at least once during each episode.



Reference torque sample

Reference excitation current sample

Figure 3-4: Reference inputs for an exemplary episode in the CCS torque control structure

Figure 3-4 demonstrates that the amplitude of the torque reference input varies from negative nominal value to positive nominal value, with sub-episode lengths changing randomly. Similarly, the excitation current fluctuates randomly between 0 and its nominal value, with the timing of these changes also being random. Moreover, the speed will also vary, ranging from positive to negative base speed within each episode. By varying both references in this manner, the agent is exposed to realistic scenarios during training.

Average Reward and Episode Length during Training

During the training process, each episode has a maximum length of $6 \cdot 10^3$ steps. Over $75 \cdot 10^4$ training steps, the agent could encounter up to 125 episodes. However, in the initial stages, the agent must learn to avoid termination and violation actions by experiencing them at least once, leading to episode resets. Consequently, it is expected that the agent will complete fewer than the maximum number of episodes, as the environment resets each time an episode is terminated early.

The average reward and episode length during the training process, as shown in figure 3-5, are calculated and averaged as discussed in section 2.3. In the early stages, both the reward and episode length are low, as the agent deals with termination and violation cases,

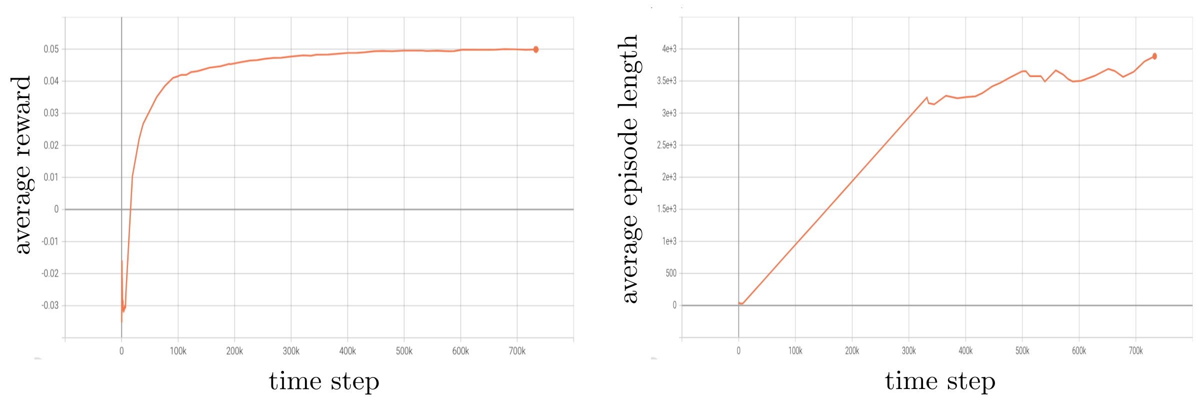


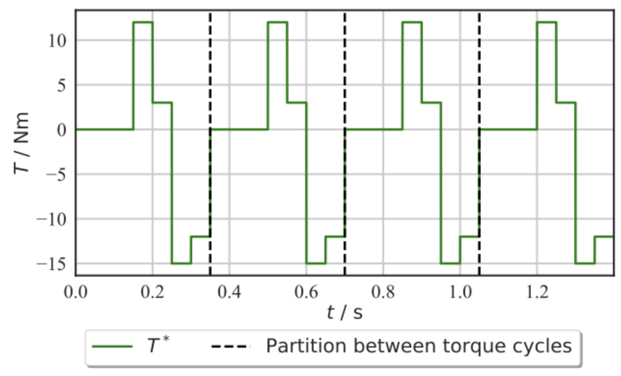
Figure 3-5: Average reward and episode length over the training steps

causing episodes to end prematurely and resulting in lower rewards. Over time, as the agent interacts with the environment and learns the system's dynamics, both the reward and episode length tend to stabilize, indicating that the agent has fully learned the system and is ready for validation.

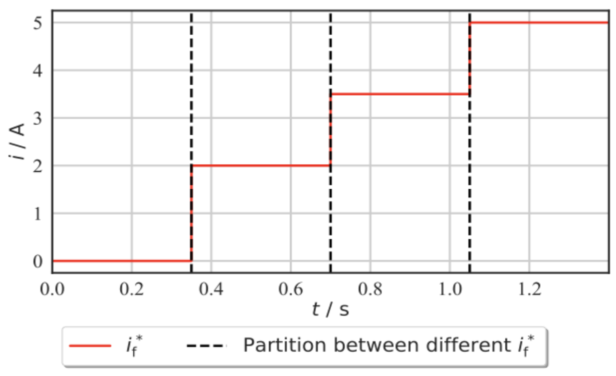
3.4 Validation and Performance Analysis

After completing $75 \cdot 10^4$ time steps of training, it is assumed that the agent has sufficiently learned the system dynamics and is ready to be deployed in a different environment. To evaluate its performance, the environment with reference torque, reference excitation and speed different from the training environment was created to test the agent's ability to handle various operating points distinct from those encountered during training. In this section, two validation cases were conducted to cover a range of operational scenarios, with the reference inputs shown in figure 3-6.

The validation scenarios involve two distinct cases: case 1 tests the agent at a negative high speed of $\omega_{\text{mech}} = -200 \text{ s}^{-1}$, and case 2 at a negative low speed of $\omega_{\text{mech}} = -20 \text{ s}^{-1}$. To highlight the differences between these operations, the speed ratio is set to a factor of 10 between the two cases. Although the agent was trained with a field current that changed only once per episode, the validation environment introduces four different field current levels within a single episode. For each field current level, the agent experiences a full torque cycle,



Reference torque cycles



Reference excitation current as step-wise

Figure 3-6: Reference inputs to a validation environment

as depicted in the figure 3-6. The validations with positive low speed and positive high speed are shown in the appendix B.

Analysis of the results

The validation results for both case 1 and case 2 are presented in figures 3-7 and 3-8. A clear distinction between the two cases is evident in the applied stator voltage. As seen in equation (1.15), higher speeds demand higher voltages during linear region operation. Consequently, u_{sq} in case 1 is significantly higher than in case 2, by approximately 10 times, reflecting the similar speed ratio. Additionally, as noted in the PI tuning section, the spike in excitation voltage u_{f} is substantial, necessary to drive the field current i_{f} back to steady state, which motivated the removal of the voltage limit. This spike occurs in response to disturbances caused by transients in the d-axis current due to the strong coupling between axes. However, during steady-state operation, u_{f} remains within the $\pm U_{\mathrm{dc}}$ range.

In both cases, the d-axis current i_{sq} closely follows the reference torque behavior, while the d-axis current is maintained above or slightly below zero, as specified in the reward design to enhance efficiency. This indicates that the agent has learned the system's dynamics well, ensuring that the machine operates with stator current on the right side of the dq-plane, while the q-axis current moves between positive and negative values as required by the reference torque. The excitation current tracks its reference exponentially, thanks to the low-pass filter, preventing disturbances in the stator current.

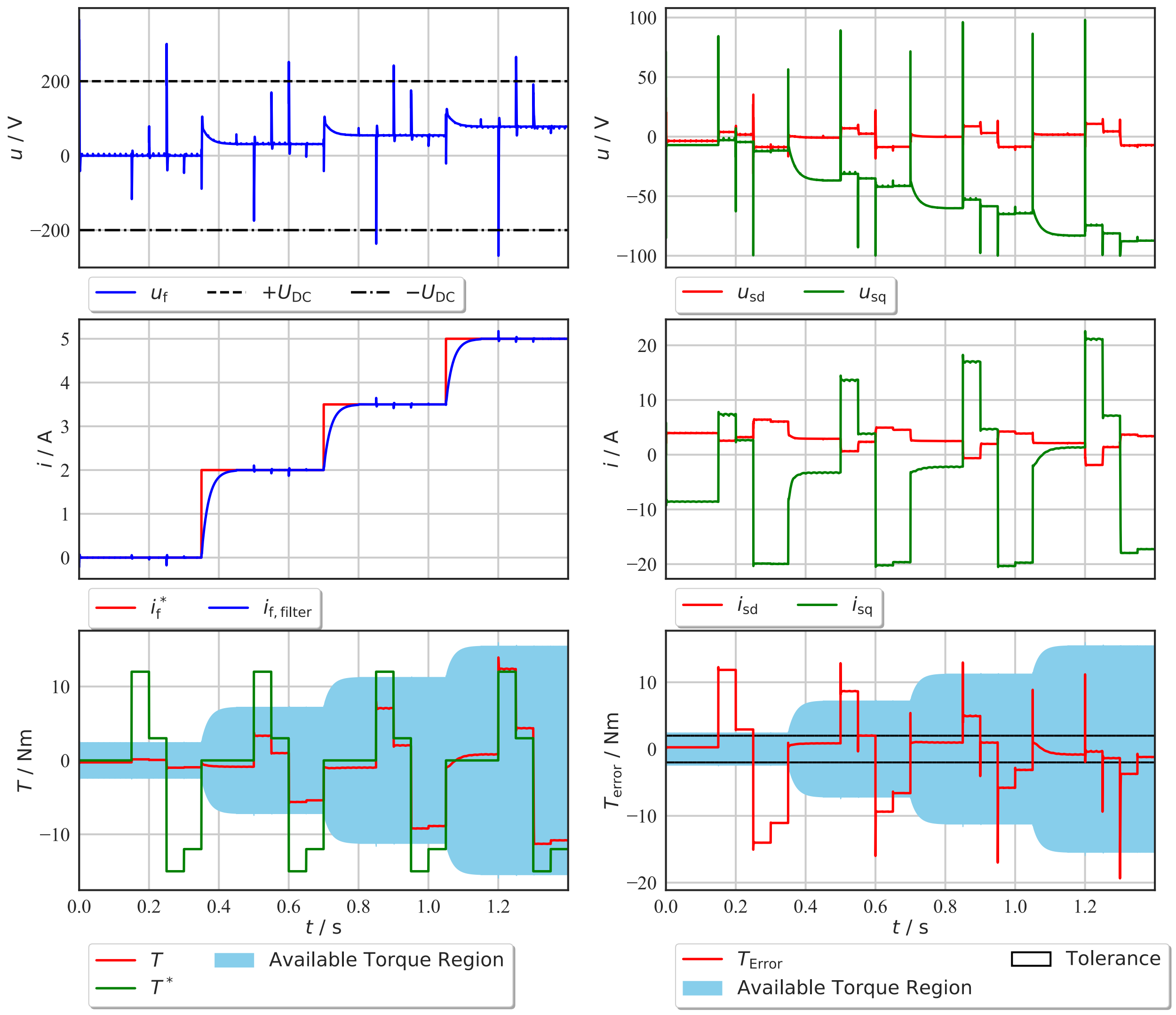


Figure 3-7: Validation results of the agent at a speed of $\omega_{\text{mech}} = -200\text{s}^{-1}$

To evaluate the agent's performance, it is crucial to analyze the torque tracking characteristics and the associated error. The blue-shaded area in the torque tracking curve represents the available torque region for the given excitation current. When the excitation current is zero, the reference torque exceeds the available electromagnetic torque, resulting in a high error in this region due to the physical infeasibility of tracking the reference. At medium field current levels (2A - 3.5A), the agent can track low reference torque with the error less than the defined tolerance, while higher reference torques fall outside the available region, making them impossible to track. Finally, with nominal field current supplied, the agent successfully tracks various reference torques within the tolerance limits. Therefore, a closer examination of the torque error curves reveals that the error remains within the defined tolerance when the reference torque is within the available torque region, while it increases when tracking the reference is physically infeasible.

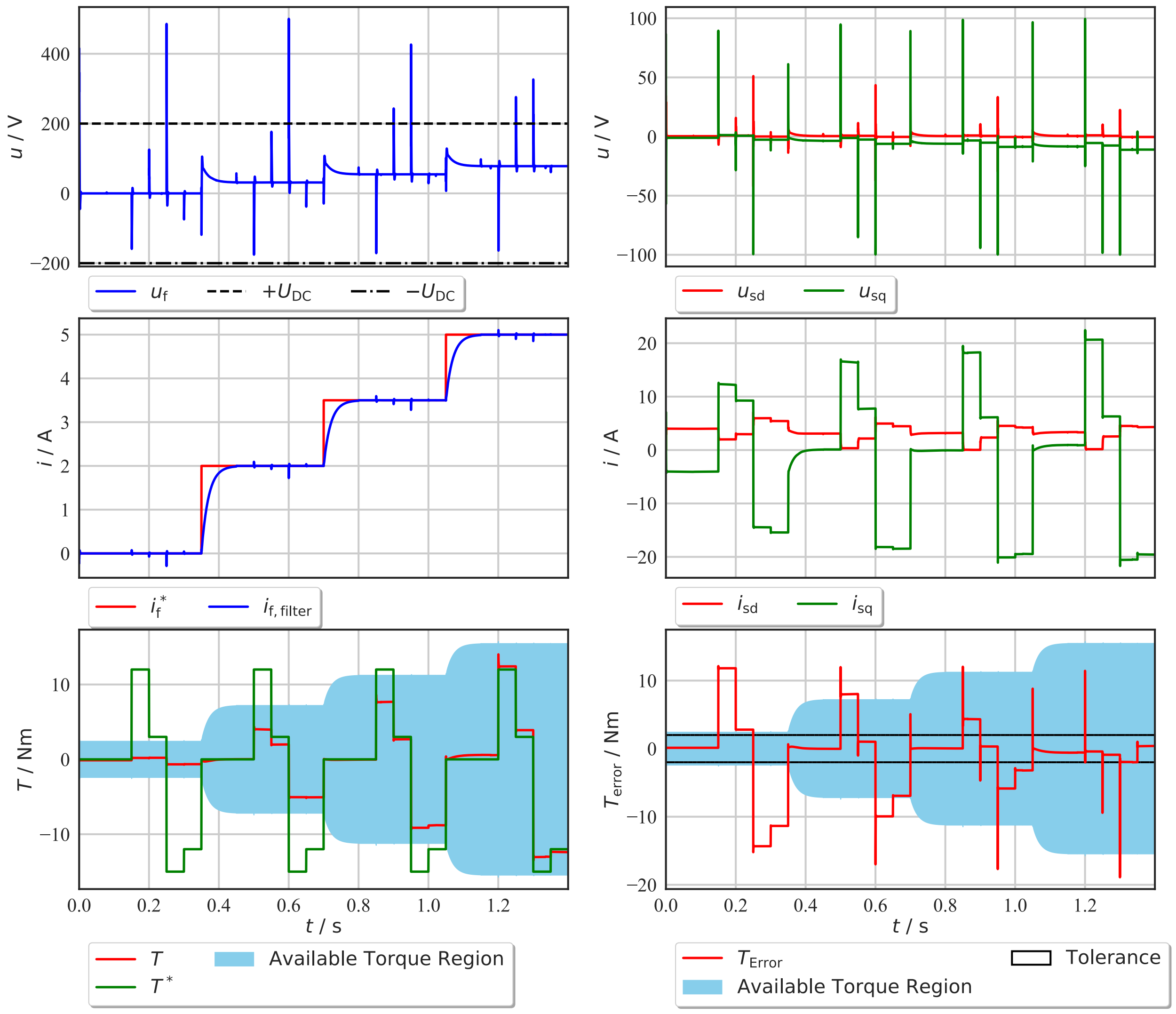


Figure 3-8: Validation results of the agent at a speed of $\omega_{\text{mech}} = -20\text{s}^{-1}$

However, even with the nominal excitation current, the torque tracking error at a high speed of $\omega_{\text{mech}} = -200\text{s}^{-1}$ slightly exceeds the defined tolerance. As seen in equation (1.19), high-speed operation demands more voltage during transients, making it inherently more challenging for the agent to control the machine at high speed compared to low speed, consequently, the tracking error remains within the limit at low speeds under nominal excitation current. However, in steady state, the agent should be able to track the high reference torque even during high-speed operation, as long as it remains within the available torque region. This discrepancy may be attributed to the agent not fully mastering the operation of the machine at high speeds, possibly due to a lack of sufficient high-speed scenarios during training.

Validation with excitation current as a ramp input

It is particularly interesting to validate the agent's performance with the excitation current applied as a ramp input. Given that the agent was trained with an exponentially changing field current, it would be notable if it could effectively manage a completely different input style. In this environment, the low-pass filter is removed, and the excitation current is gradually increased from zero to its nominal value in a ramp fashion, while the reference torque remains constant at the nominal value throughout the validation period.

As the excitation current increases slowly, the coupling effect on other axes is very low, with the excitation voltage exceeding the limit only once at the start. Furthermore, as shown in figure 3-9, the way the agent provides the actions (u_{sd}, u_{sq}) is noteworthy. During the transient phase of the excitation current, it is not practically feasible to track the reference torque because the available torque region is lower than the reference. In this scenario, the agent does not attempt to apply the maximum stator voltage to track the reference. Additionally, the d-axis current remains above zero throughout the period, as defined in the reward design, even during the transient.

Finally, when the excitation current stabilizes, the agent also stabilizes u_{sq}, successfully tracking the reference torque with very low error. The error remains close to zero, though not exactly zero, due to the design of the reward function as discussed below.

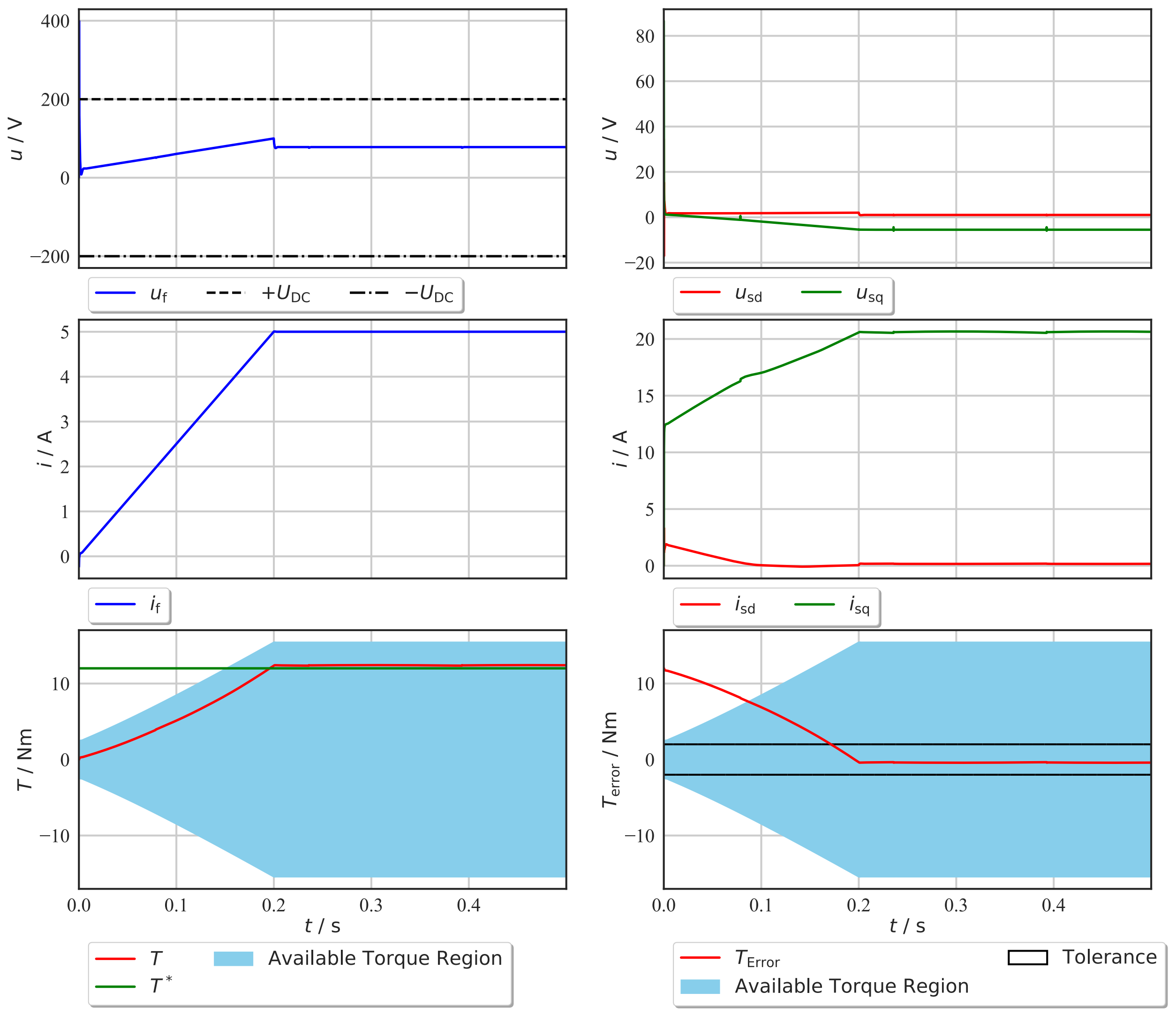


Figure 3-9: Validation results of the agent with i_f^* as a ramp input at a speed of -20s^{-1}

According to reward design, the agent attempts to track the reference torque with an error that remains within the specified tolerance. Once the error is within this limit, the agent shifts its focus from further reducing the error to reducing copper losses with stator current. This shift is evident in figure 3-9, where the agent refrains from increasing the stator current to further minimize the error, as the error is already within acceptable limits as defined in region $\mathbb{A}$ of the reward design.

$$\text{Region } \mathbb{B}, \quad \text{if} \quad (i_{s,k} < i_{nom}) \quad \text{and} \quad (i_{sd,k} > 0) \quad \text{and} \quad (|T_k^* - T_k| > T_{tol}) :$$

$$\Rightarrow \quad r_k = \left(1 - \left|\frac{T_k^* - T_k}{2T_{\text{lim}}}\right|\right) \frac{1-\gamma}{2} \tag{3.15}$$

$$\Rightarrow \quad r_k \in [0, 0.05] \tag{3.16}$$

$$\text{Region } \mathbb{A}, \quad \text{if} \quad (i_{\text{s},k} < i_{\text{nom}}) \quad \text{and} \quad (i_{\text{sd},k} > 0) \quad \text{and} \quad (|T_k^* - T_k| < T_{\text{tol}}):$$

$$\Rightarrow \quad r_k = \left(1 - \frac{i_{\text{s},k}}{i_{\text{lim}}}\right) \frac{1-\gamma}{2} + \frac{1-\gamma}{2} \tag{3.17}$$

$$\Rightarrow \quad r_k \in [0.05, 0.1] \tag{3.18}$$

Region $\mathbb{A}$ and $\mathbb{B}$ of the reward design from section 3.2.2 are numerically mentioned above. The agent prioritizes efficiency by reducing or maintaining the current level, demonstrating the MTPC strategy by stator current once the error is within the defined tolerance. Consequently, the maximum reward of 0.1 does not correspond to minimal error but is instead related to the applied stator current. Despite the brief training of $75 \cdot 10^4$ time steps, the agent's performance is commendable.

3.5 Key Takeaways of the Chapter

This chapter represents a critical step towards achieving optimal torque control of EESM using an RL-based controller. A PI-assisted RL controller was employed to facilitate a smooth transition from current control to torque control in this chapter. The primary focus was on adapting the PMSM reward design from [8] to enable the EESM to behave similarly to a PMSM under constant excitation current conditions. To this end, an idealized environment was created by removing the excitation voltage limit and employing a high working bandwidth for the PI controller. Although this setup is not realistic for physical implementation, it provides valuable insights for future work. Specifically, the results indicate that an RL agent can manage the complex dynamics of an EESM for torque control, track the reference torque by manipulating stator current, and adapt to changes in excitation current, even when it is not controlled by the agent. In the next chapter, the torque control will be further developed using a fully RL-based controller, with the removal of the idealized setup.

Chapter 4

Torque Control with Reinforcement Learning based Excitation Circuit

In chapter 3, it was demonstrated that the DDPG agent can effectively control the EESM in a manner similar to a PMSM, with the excitation current controlled a PI controller. In this chapter, the focus shifts to a more comprehensive approach where the agent independently manages the torque control of the EESM. This means that, unlike in the previous analysis where the agent provided only the d- and q-axes voltages, it will now control all three currents by applying three voltages to track the reference torque. Consequently, the overall control structure has been modified from the previous setup, and the new configuration for torque control, including an RL-based excitation circuit, is presented in figure 4-1.

As shown in figure 4-1, the reference torque is the only input parameter in this control structure. The agent generates three actions (u_{sd}, u_{sq}, u_{f}) and interacts with the environment to learn and adapt over time. The observation state for the agent in this control structure is as follows:

$$\boldsymbol{o}_k = \left[\tilde{i}_{\mathrm{sd},k} \;\; \tilde{i}_{\mathrm{sq},k} \;\; \tilde{i}_{\mathrm{f},k} \;\; \tilde{\omega}_{\mathrm{mech},k} \;\; \cos\left(\epsilon_{\mathrm{el},k}\right) \;\; \sin\left(\epsilon_{\mathrm{el},k}\right) \;\; \tilde{T}^{*}_{k} \;\; \tilde{u}_{\mathrm{sd},k} \;\; \tilde{u}_{\mathrm{sq},k} \;\; \tilde{u}_{\mathrm{f},k}\right]. \tag{4.1}$$

As discussed previously, these states must be normalized, with the exception of $\cos\left(\epsilon_{\mathrm{el},k}\right)$ and $\sin\left(\epsilon_{\mathrm{el},k}\right)$. The normalization range remains consistent with the values mentioned in

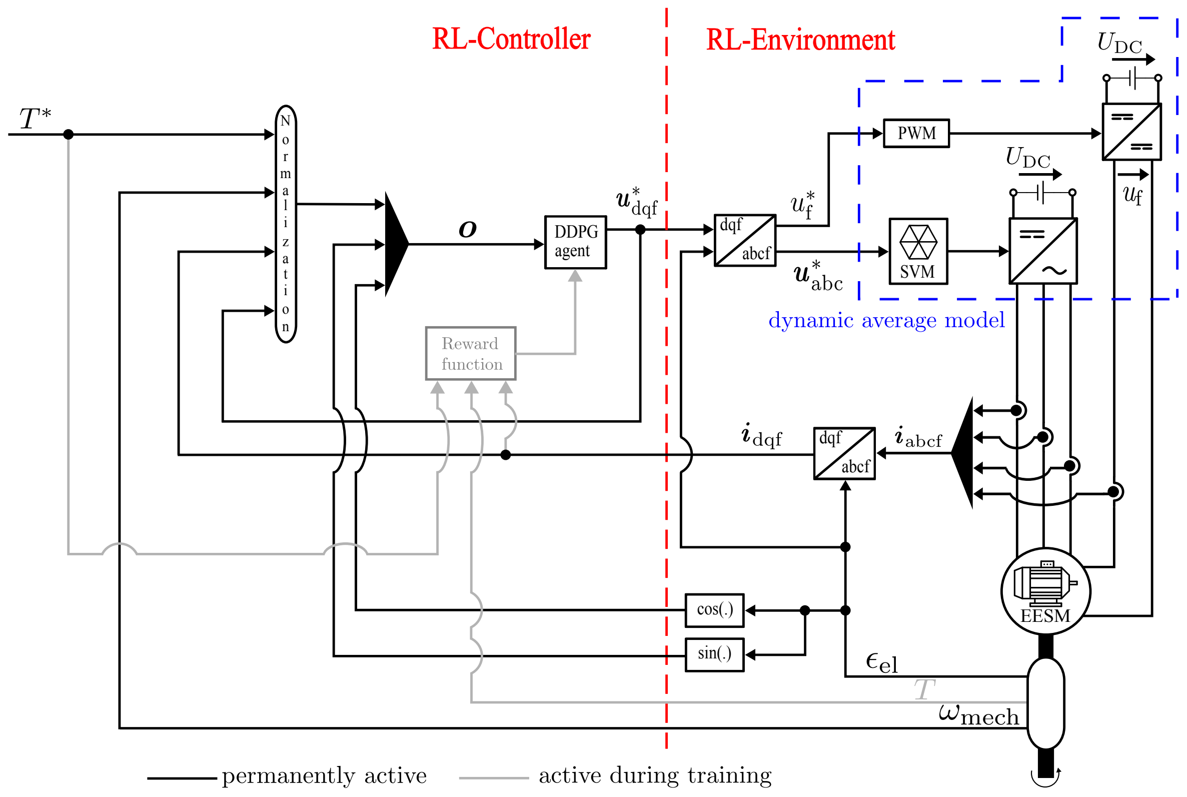


Figure 4-1: Schematic of the overall CCS torque control structure with RL-based excitation circuit

earlier chapters. The agent's working principle is also similar to that described earlier: without any predefined control knowledge, the agent begins by applying random actions to the RL environment and learns through the feedback it receives via the reward function. Consequently, the reward function is crucial in shaping the agent's performance. This chapter is divided into two main sections: performance priority control and efficiency priority control, which differ based on the reward design.

4.1 Performance Priority Control

In the control structure illustrated in figure 4-1, the excitation circuit is also controlled by the DDPG agent, replacing the previous PI controller. To ensure a smooth transition from the PI-assisted RL control to a fully RL-based control structure, the reward design from the previous chapter will be applied, with updates only to the safety constraints. This approach allows the DDPG agent the flexibility to adjust the field current as needed, as long as it effectively tracks the reference torque.

4.1.1 Reward Design for Performance Priority Control

The reward design approach for performance priority control is shown in figure 4-2. While the excitation current does not directly influence the reward design, it is essential to incorporate its limitations into the existing reward structure to address safety concerns. Consequently, both the over-current operation of the stator circuit and the rotor circuit must be restricted to prevent overheating. Additionally, negative i_f would negatively impact efficiency and must be constrained. Therefore, operations in the excess current region $\mathbb{E}$ and the unfavorable efficiency region $\mathbb{C}$ will align with the previous reward design.

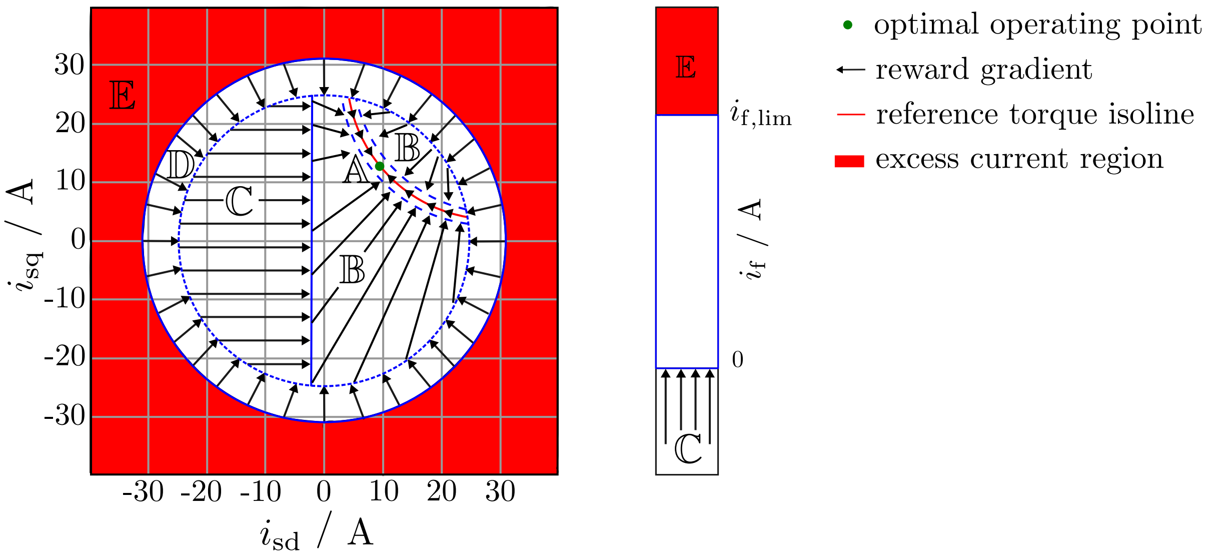


Figure 4-2: Schematic of the reward design approach for performance priority control

1: Excess current region, $\mathbb{E}$

This region consists of two parts: one where the stator current exceeds its limit and the other where the rotor current exceeds its limit. If either the stator or rotor current enters this region, it is considered a violation of the constraints, leading to the termination of the episode and the initiation of a new one. As in the previous design, the penalty reward will correspond to the extent to which the current exceeds the excess current region.

1.1: Region $\mathbb{E}$ in stator circuit

When the stator current enters this region, the following penalty reward will be applied.

$$\text{if} \quad (i_{\mathrm{s},k} > i_{\mathrm{s,lim}}) \Rightarrow \quad r_k = \left(-0.1 - \frac{i_{\mathrm{s},k}}{i_{\mathrm{s,lim}}}\right) \tag{4.2}$$

1.2: Region $\mathbb{E}$ in rotor circuit

When the excitation current enters this region, a corresponding penalty reward will be applied.

$$\text{if} \quad (i_{\mathrm{f},k} > i_{\mathrm{f,lim}}) \Rightarrow \quad r_k = \left(-0.1 - \frac{i_{\mathrm{f},k}}{i_{\mathrm{f,lim}}}\right) \tag{4.3}$$

2: Unfavorable efficiency region, $\mathbb{C}$

This region also consists of two parts: one where the d-axis current reduces efficiency and another where the f-axis current negatively impacts efficiency. With the agent now controlling the excitation current, both negative d-axis current and negative f-axis current can reduce efficiency, as discussed in equation (1.26). However, it is crucial to allow the agent some flexibility during the learning phase to understand why operating in this region results in a negative reward. Thus, if the agent fails to operate the machine on the right side of the dq-plane with a field current greater than zero, the following reward will be applied.

2.1: Region $\mathbb{C}$ in stator circuit

When the agent operates the machine with negative d-axis current, the corresponding reward will be applied.

$$\text{if} \quad (i_{\mathrm{s},k} < i_{\mathrm{s,nom}}) \quad \text{and} \quad (i_{\mathrm{sd},k} < 0) \Rightarrow \quad r_k = \left(1 + \frac{i_{\mathrm{sd},k}}{i_{\mathrm{s,lim}}}\right)\frac{1-\gamma}{2} - \frac{1-\gamma}{2} \tag{4.4}$$

$$\Rightarrow \quad r_k \in \left[-\frac{1-\gamma}{2}, \quad 0\right] \tag{4.5}$$

2.2: Region $\mathbb{C}$ in rotor circuit

Similarly, when the agent operates the machine with negative f-axis current, the respective reward will be applied.

$$\text{if} \quad (i_{\mathrm{f},k} < 0) \Rightarrow \quad r_k = \left(1 + \frac{i_{\mathrm{f},k}}{i_{\mathrm{f,lim}}}\right)\frac{1-\gamma}{2} - \frac{1-\gamma}{2} \tag{4.6}$$

$$\Rightarrow \quad r_k \in \left[-\frac{1-\gamma}{2}, \quad 0\right] \tag{4.7}$$

Apart from updating these two regions, the remaining regions ($\mathbb{A}$, $\mathbb{B}$, $\mathbb{D}$) will retain the same design as before. Similarly, the prioritization of operating regions remains unchanged, with region $\mathbb{A}$ being the least important and region $\mathbb{E}$ the most important concerning the safety.

4.1.2 Training the DDPG agent

To balance the trade-off between training time and the agent's performance, the training duration is slightly extended compared to the previous setup, as the agent must manage more complex tasks in this control structure. Consequently, a total of $90 \cdot 10^4$ training steps are utilized, corresponding to a real-time simulation duration of $1 \cdot 10^{-4} \times 90 \cdot 10^4 = 90$ seconds. However, due to the computational demands, the training process required 123 minutes on the workstation to run 90 seconds of real-time operation and update the training network. The hyper-parameters and ANN architecture for this setup are selected as outlined below.

Overview of Selected Hyper-parameters Set

With the transition from PI-assisted to fully RL-based torque control, the agent is now responsible for handling an additional action, u_f. As a result, it is necessary to re-select and adjust the hyper-parameters to estimate optimal tuning for this new control structure. These hyper-parameters differ from the previous set to accommodate the expanded control responsibilities of the agent. The hyper-parameters used in this RL-based optimal torque control are listed below.

Symbol	**Description**	**Selected set**
γ	discount factor	0.9
$\mathcal{D}_b$	mini-batch size	128
$\mathcal{D}$	replay buffer size	$91 \,.\, 10^4$
K	maximum episode length	$1 \,.\, 10^4$
M	total training steps	$90 \,.\, 10^4$
α_actor	actor learning rate	$1 \,.\, 10^{-5}$
α_critic	critic learning rate	$1 \,.\, 10^{-4}$
$F_\mathrm{prediction}$	prediction networks update parameter	1 time step
N_train	memory warm-up	$1 \,.\, 10^4$
ν	action noise	see figure 4-3
T_tol	torque control tolerance	0.1%
f_s	sampling frequency	10 kHz

Table 4.1: Selected hyper-parameters set for performance priority torque control

The primary differences from the previous setup include the learning rate, episode length, and action noise. A smaller, constant learning rate is utilized to prevent the ANNs calculations from deviating from the optimal solution, and this rate remains unchanged throughout the entire training period. Additionally, a longer episode length is adopted, as it takes approximately 2000 steps for the excitation current to reach a steady state, as observed in chapter 2. This extended duration allows the agent enough time not only to learn the system's dynamics but also to minimize errors within the defined tolerance.

Regarding exploration noise, Gaussian noise is utilized in this control structure as well, with zero mean values. The standard deviation of the noise varies over the course of training, as illustrated in figure 4-3.

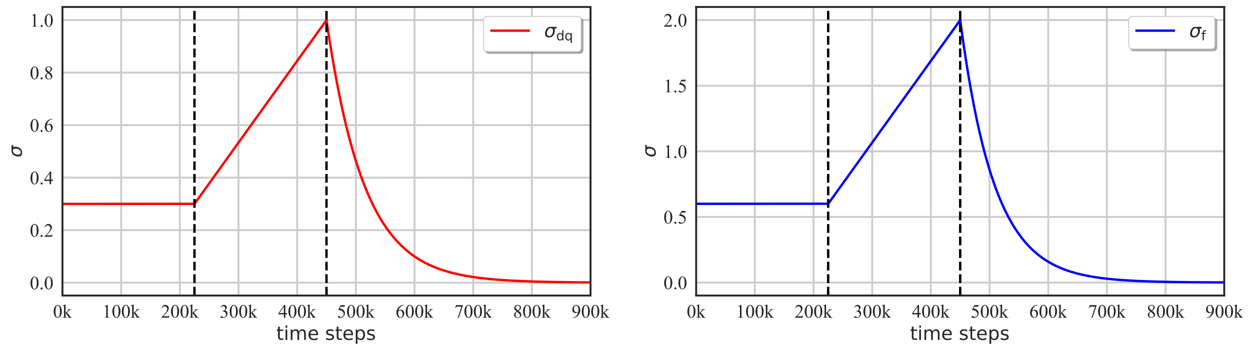


Figure 4-3: Variation of the standard deviation of the Gaussian distribution over time steps for dq- and f-actions

Given that the rotor voltage range is different from that of the stator voltage, with $\boldsymbol{u}_{\mathrm{dq}} \in$ [-100, 100] and $u_{\mathrm{f}} \in$ [-200, 200], it is appropriate to apply action noise with different standard deviations. Specifically, the standard deviation, σ for the f-axis action is set to be twice as large as that for the dq-axes actions. Despite this variation in standard deviation, the overall behavior of the action noise follows the same pattern for all three actions, progressing through three distinct phases: the initial stage, exploration stage, and exploitation stage, as discussed in the section 2.3.

Artificial Neural Network Architecture

Once the estimated optimal hyper-parameters are established, the next step is to customize the ANN architecture. Given that the agent now has to manage additional actions, a larger network is required compared to the previous setup. To balance the trade-off between computational load and adequate network size, the ANN is configured as listed in table 4.2.

Network	**Layer Type**	**Neurons**	**Activation Function**	**Definition**
Actor	input layer	10	-	-
	hidden layer 1	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 2	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 2	128	LeakyReLU	$\max(0.2x, x)$
	output layer	3	Tanh	$\tanh(x)$
Critic	input layer	13	-	-
	hidden layer 1	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 2	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 3	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 4	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 4	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 4	128	LeakyReLU	$\max(0.2x, x)$
	hidden layer 5	128	LeakyReLU	$\max(0.2x, x)$
	output layer	1	Linear	x

Table 4.2: Artificial neural network configuration for CCS torque control

To accommodate the fully RL-based torque control, an additional hidden layer is added to the actor network, and two more hidden layers are incorporated into the critic network.

Pattern of Reference

As illustrated in figure 4-1, the torque reference is the only input to the control structure and is applied as a step-wise signal, with its amplitude and sub-episode length changing randomly, as shown in the figure 4-4.

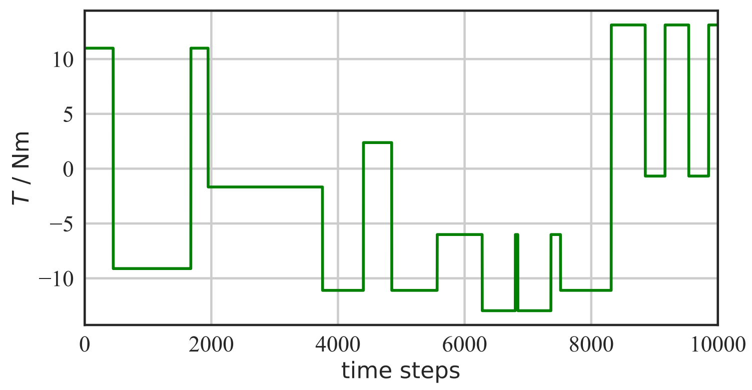


Figure 4-4: Reference input for an exemplary episode in the CCS torque control structure

Average Reward and Episode Length during Training

During the $90 \cdot 10^4$ training steps, with a maximum episode length of $1 \cdot 10^4$ steps, the agent could numerically encounter up to 90 episodes. However, as discussed in the previous chapters, it is likely that the agent experiences fewer full-length episodes due to early terminations. As depicted in figure 4-5, both the average reward and episode length increase exponentially and linearly over time, eventually flattening in the later stages, indicating that the agent has sufficiently learned the control strategy. Nevertheless, while the agent appears ready for validation in a different environment, further improvements in rewards might be achievable with extended training.

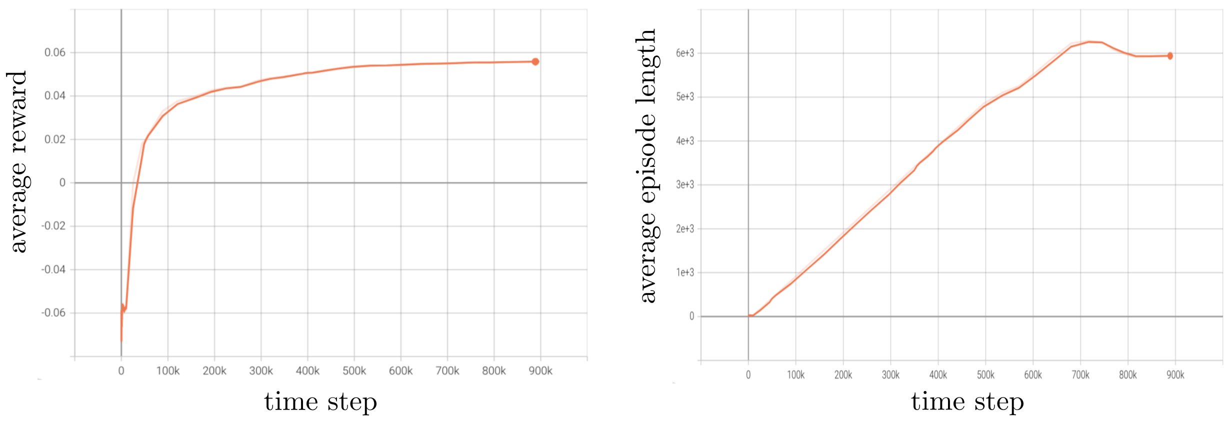


Figure 4-5: Average reward and episode length over the training steps

4.1.3 Validation and Performance Analysis

To assess the agent's ability to adapt to different conditions, a different environment was created, with the behavior of the torque reference illustrated in figure 4-6.

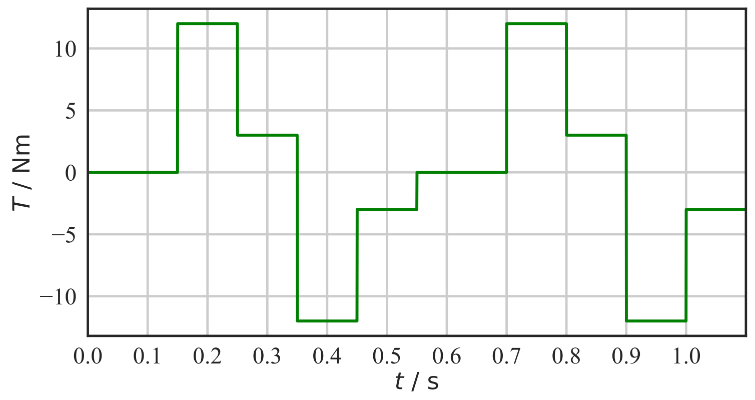


Figure 4-6: Reference torque to a validation environment

Two validation cases were conducted at positive low speed and negative low speed with a full torque cycle to cover all operating points. However, for the sake of brevity, only the validation case at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$ is discussed here, with the other case provided in the appendix C. The validation results for the selected case are presented in figure 4-7.

Analysis of the results

At a glance, it is clear that the agent adapts well to the changes in an environment, as it successfully tracks the reference torque with an error below the predefined tolerance. It is important to note that the available torque region remains nominal throughout the analysis, as the agent is allowed to apply the excitation current freely, as long as it is within the safety region.

A closer examination of the graphs reveals further details. In the actions graph, the agent-provided values for u_{sd}, u_{sq} and u_{f} demonstrate that the stator voltage remains within the allowable range $u_{\text{stator}} \in [-100, 100]$ and the rotor voltage also stays within its specified limits $u_{\text{rotor}} \in [-200, 200]$ as discussed in the power electronic converter section 1.3.5.

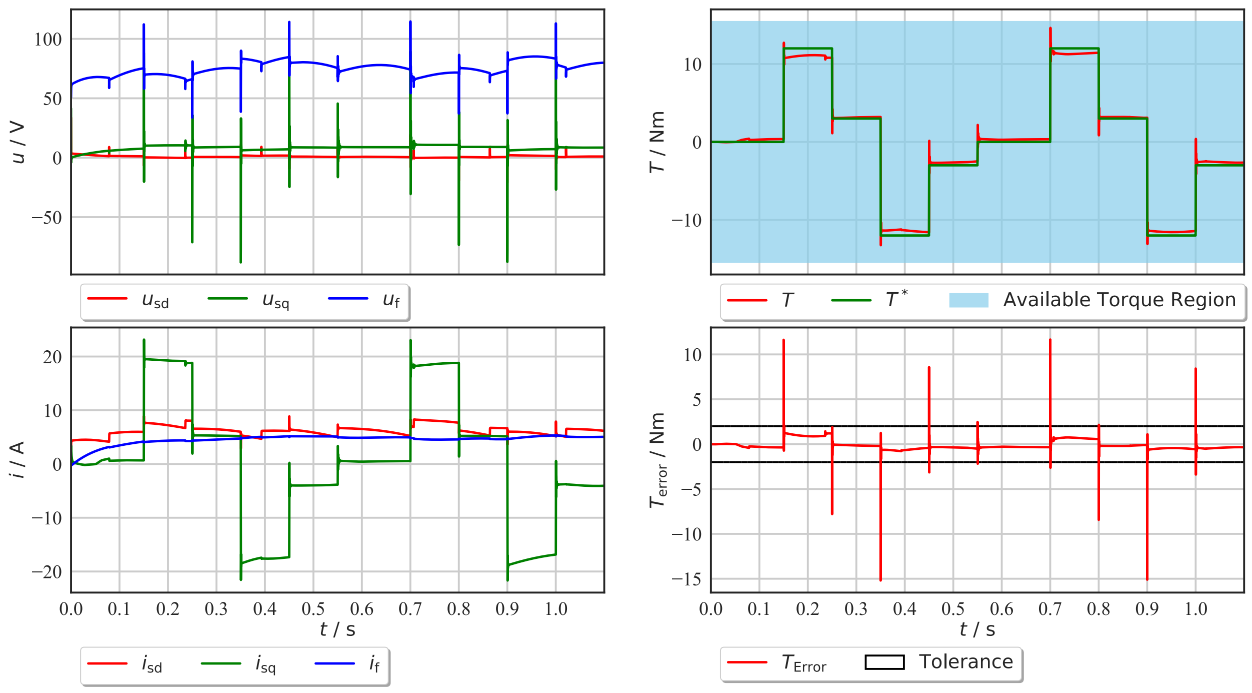


Figure 4-7: Validation results of performance priority control at a speed of $\omega_{\mathrm{mech}} = 20\mathrm{s}^{-1}$

Furthermore, one noteworthy observation in the currents graph is that the excitation current remains at its nominal value after the transient, regardless of changes in the reference torque. The agent successfully tracks the reference torque by primarily manipulating the stator current. However, all three currents respond appropriately to the applied actions, aligning well with the reward design. None of the currents enter the excess current region $\mathbb{E}$, and the agent ensures that the stator current stays below the short-term over-current region $\mathbb{D}$. Additionally, i_{sd} and i_{f} remain above zero, as required by the efficiency considerations in region $\mathbb{C}$. Finally, the agent adjusts the currents effectively to track the reference torque, maintaining the error within the tolerance defined in the desired operating region $\mathbb{B}$.

Once the agent achieves torque tracking within the desired operating region, it shifts focus to implementing the MTPC strategy with stator current. As seen in the torque tracking graph, the agent no longer prioritizes minimizing the error, as it is already within acceptable limits. Instead, the agent aims to reduce or maintain the stator current while tracking the reference torque, emphasizing efficiency. This behavior is further evidenced in figure 4-8, where the reward corresponds to the applied stator current once the error is within tolerance.

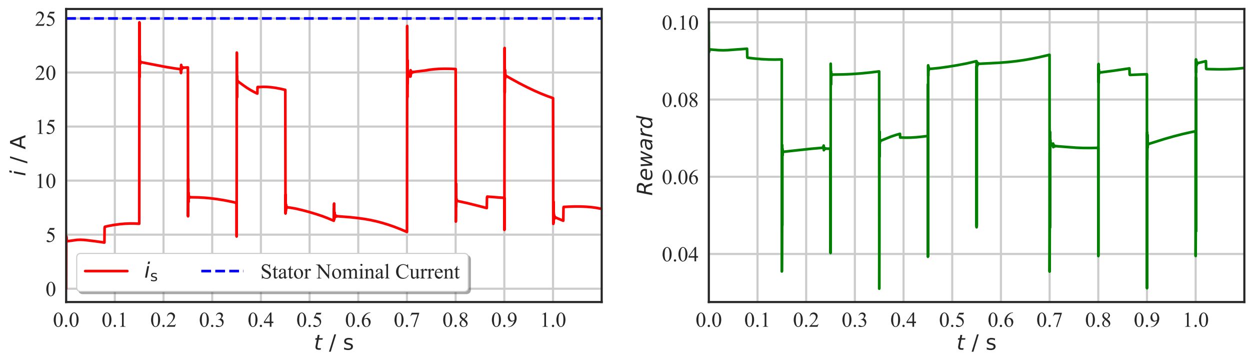


Figure 4-8: Reward corresponds to reference torque isoline region $\mathbb{A}$

For instance, when the stator current is high (around 20A) between 0.15 and 0.25 seconds of simulation time, the reward decreases to approximately 0.7. Conversely, between 0.45 and 0.07 seconds, the stator current is relatively low, and the reward increases to nearly 0.09, close to the maximum achievable reward of 0.1. This pattern reflects the machine operating within region $\mathbb{A}$, following the MTPC strategy with stator current. Therefore, considering the agent's brief training time steps of $90 \cdot 10^4$, its performance is highly commendable.

4.2 Efficiency Priority Control

As discussed earlier, since the excitation current was not included in the reward function, the agent maintained it at a nominal value throughout the operation, potentially impacting efficiency. This section now shifts focus to efficiency priority control, where the excitation current is incorporated into the reward design with an emphasis on maximizing efficiency. The following discussion will explore the critical roles that both stator and rotor currents play in enhancing overall efficiency.

4.2.1 Insight to Efficiency

To effectively integrate efficiency into the reward design for efficiency priority control, it is essential to take into consideration the method of efficiency calculation, as outlined below:

$$\eta = \frac{P_{\text{out}}}{P_{\text{in}}}, \tag{4.8}$$

where $P_{\text{out}} \in \{P_{\text{ele}}, P_{\text{mech}}\}$ and $P_{\text{in}} \in \{P_{\text{ele}}, P_{\text{mech}}\}$ are considered in order to cover both motor and generator operations. The electrical power can be calculated as follows:

$$\begin{aligned} P_{\text{el,stator}} &= \frac{3}{2}\left(u_{\text{sd}} i_{\text{sd}} + u_{\text{sq}} i_{\text{sq}}\right), \\ P_{\text{el,rotor}} &= u_{\text{f}} i_{\text{f}}, \\ P_{\text{ele}} &= P_{\text{el,stator}} + P_{\text{el,rotor}}. \end{aligned} \tag{4.9}$$

Then, the mechanical power can also be evaluated as follows:

$$P_{\text{mech}} = T_{\text{em}} \omega_{\text{mech}}. \tag{4.10}$$

In this approach, the electrical power will be considered as input power if it is greater than the mechanical power, output power, as the other way around, the mechanical power becomes input power when it is greater than the electrical power. However, this approach is only accurate in steady-state operation. During transients, some input power is temporarily stored in the motor's magnetic field, known as magnetic power, and does not immediately convert to mechanical output.

As an alternative approach, calculating efficiency based on ohmic losses is straightforward and applicable to both transient and steady-state conditions. However, this method requires knowledge of stator and rotor resistance values (machine parameters), which vary with temperature. Given that the primary objective of this analysis is to develop a data-driven controller that does not rely on machine parameters, using resistance values is not preferred. Instead, efficiency will be calculated using the ratio of output power to input power as shown in equation (4.8), which depends only on observable variables (voltage, current, and speed). This method applies to both motor and generator operations and will be integrated into the reward design to enhance efficiency in this analysis.

4.2.2 Reward Design for Efficiency Priority Control

In the performance priority control discussed above, the excitation current i_f was not included in the region $\mathbb{A}$ of reward design, potentially leading to significant losses. To address this, the efficiency priority control will now incorporate both the stator current and excitation current into the reward design, with a focus on optimizing efficiency. While the safety and operational constraints defined in regions $\mathbb{E}$, $\mathbb{D}$, $\mathbb{C}$ and $\mathbb{B}$ from the performance priority control will remain unchanged, the reward associated with region $\mathbb{A}$ will be revised to emphasize efficiency.

1: Efficiency enhancement in region, $\mathbb{A}$

$$\text{if} \quad (|T_k^* - T_k| < T_\mathrm{tol}) \quad \Rightarrow \quad r_k = \eta_k \left(\frac{1-\gamma}{2}\right) + \frac{1-\gamma}{2} \tag{4.11}$$

$$\Rightarrow \quad r_k \in \left[\frac{1-\gamma}{2}, \quad 1-\gamma\right] \tag{4.12}$$

In the earlier design, region $\mathbb{A}$ focused solely on minimizing stator current, allowing the agent to apply the excitation current i_f without restriction. Under the efficiency priority control, efficiency calculations are now integrated into the reward structure, ensuring a more balanced approach. The updated reward design is straightforward: higher efficiency results in higher rewards. By incorporating efficiency directly into the reward function, the agent's control strategy becomes more comprehensive, targeting efficiency improvements across all three currents.

4.2.3 Training the DDPG Agent

For this analysis, $90 \cdot 10^4$ training steps were utilized, corresponding to a physical simulation duration of 90 seconds at a sampling frequency of 10 kHz. However, since the training was conducted on an asynchronous simulation and given the computational demands, it took 124 minutes to update 90 seconds of training on the workstation. The selected hyper-parameters and ANN configuration are discussed below.

Overview of Selected Hyper-parameters Set

The overall control structure remains consistent between performance and efficiency priority controls, meaning that most aspects, such as hyper-parameters, ANN architecture, and torque reference behavior, remain unchanged. The primary differences lie in the updated reward design, which now incorporates efficiency considerations as discussed in section 4.2.2, and in the slight adjustments to the action noise, as shown in figure 4-9.

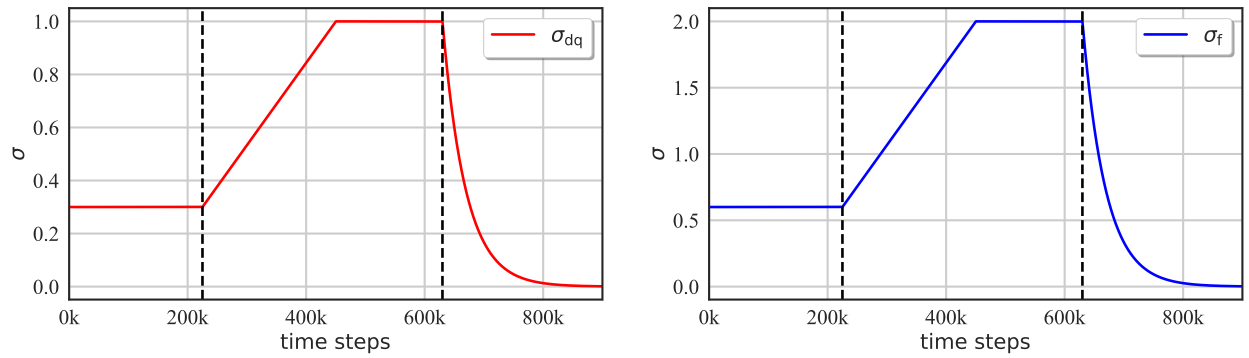


Figure 4-9: Variation of the standard deviation of the Gaussian distribution over time steps for dq- and f-actions

In the efficiency priority control, Gaussian noise with zero mean and varying standard deviation is applied, similar to the previous chapter. The standard deviation for the f-axis action remains twice that of the dq-axis actions, consistent with their respective voltage ranges. However, in this analysis, the exploration period is extended: after σ reaches its maximum for full exploration, it remains at this level for a longer duration before transitioning to the exploitation phase. This adjustment expected to ensure that the agent thoroughly explores the environment before moving on to exploit its learned strategies.

Average Reward and Episode Length During Training

The progression of the average reward and episode length throughout the training process is depicted in figure 4-10.

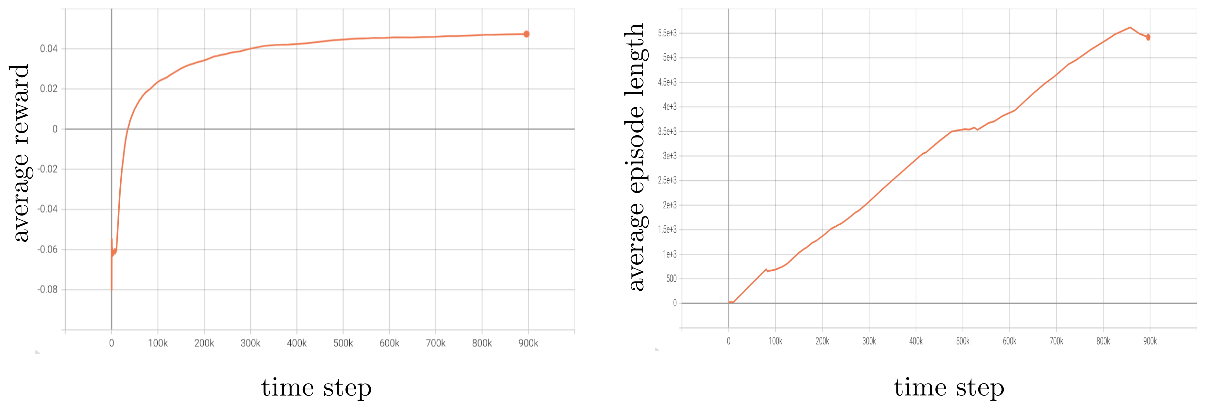


Figure 4-10: Average reward and episode length over the training steps

During the training, there is a small drop in the average episode length midway through, coinciding with the full exploration phase. This drop is expected due to the high standard deviation of the Gaussian noise used during this period, which causes the environment to reset. However, as the agent transitions from exploration to exploitation, the average episode length begins to increase steadily, while the average reward curve flattens towards the later stages of training. These trends indicate that the agent has sufficiently explored the environment, learned the necessary behaviors, and is now prepared for validation in a different environment.

4.2.4 Validation and Performance Analysis

To ensure a meaningful comparison with the performance priority control, the same validation setup is used for this analysis. The validation is conducted at the same speeds of

$\omega_{\text{mech}} = 20s^{-1}$ and $\omega_{\text{mech}} = -20s^{-1}$, covering a full torque cycle to evaluate all operating points. The results for the positive low speed are presented in figure 4-11, while the results for the negative low speed are provided in the appendix D.

Analysis of the results

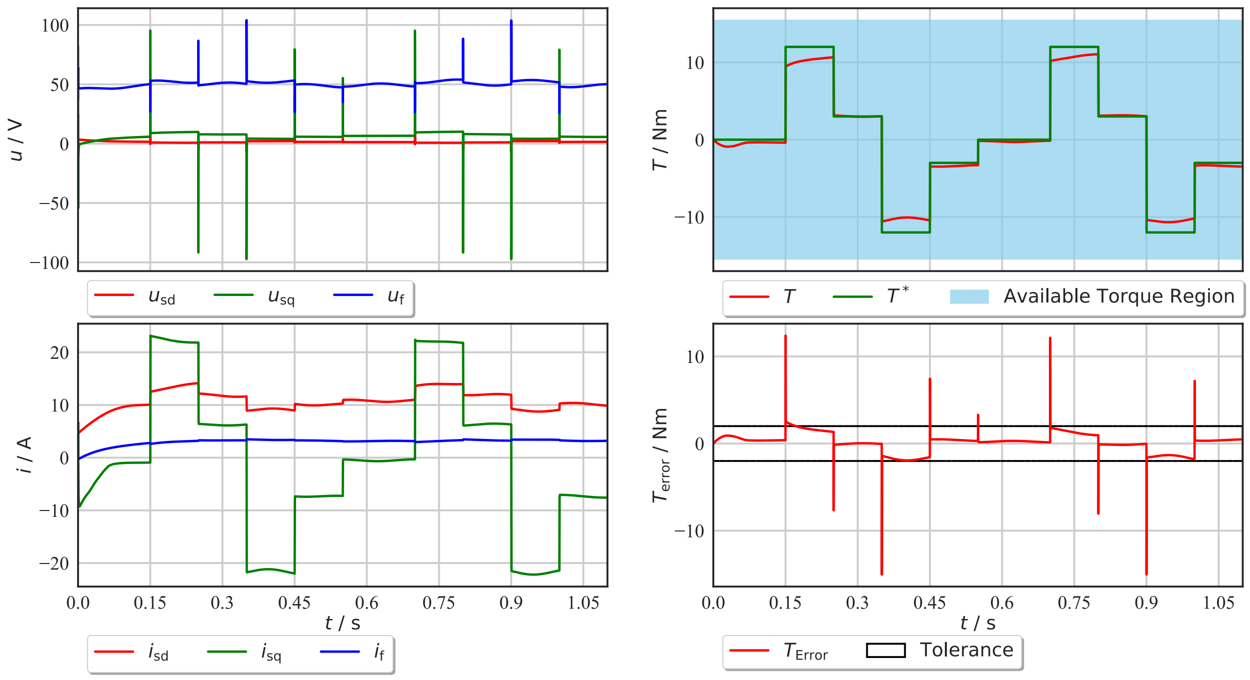


Figure 4-11: Validation results of efficiency priority control at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$

The results in figure 4-11 demonstrate that the agent effectively understands the system dynamics and tracks the reference torque. However, it is evident that the error for high torque references slightly exceeds the defined tolerance, a deviation not observed in the performance priority control (a comparison will be discussed in the following section 4.3). This discrepancy arises because the agent, in its effort to maximize the efficiency and the reward, struggles to quickly increase the excitation current to match high torque demands. This results in the agent having difficulty tracking high reference torque values.

Despite these challenges, the agent performs well in tracking low reference torque, with errors close to zero, while successfully maintaining stator and rotor voltages within the defined limits. This indicates that while the agent targets to maximize efficiency, it still manages to achieve effective control, especially in less demanding scenarios.

4.3 Comparative Analysis of Performance and Efficiency Priority Control

This section presents a comparative analysis focusing on two key aspects: performance and efficiency, as evaluated in the preceding analyses.

4.3.1 Performance Comparison

The performance comparison is based on the overall tracking error, quantified by the root mean square error (RMSE), which serves as an objective metric for assessing the accuracy of torque tracking. The RMSE can be defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(\tilde{T}_k - \tilde{T}_k^*\right)^2}, \tag{4.13}$$

where n represents the total number of time steps, and k denotes a specific time step. The RMSE for both control strategies, as depicted in figures 4-7 and 4-11, was calculated for operations conducted at a speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$, with the reference torque provided in figure 4-6. The resulting RMSE values are as follows:

$$\text{Performance Priority, RMSE} = 0.0012, \tag{4.14}$$

$$\text{Efficiency Priority, RMSE} = 0.0025. \tag{4.15}$$

These results indicate that the performance priority control strategy demonstrates superior torque tracking accuracy, with a significantly lower RMSE compared to the efficiency priority control strategy. It can therefore be concluded that the performance priority control achieves better torque tracking performance.

4.3.2 Efficiency Comparison

To evaluate the agent's efficiency in this analysis, a new scenario distinct from the previous validation setup was created. The efficiency calculation follows the ratio of output power to input power approach, as discussed in section 4.2.1, where the ratio between electrical and mechanical power provides a measure of efficiency. To thoroughly assess this, the validation environment employs a high reference torque at medium positive speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$, covering both motor and generator operations. The reference torque applied is depicted in figure 4-12, with the zero reference torque set at the start of the simulation to allow the excitation current to reach a steady state before conducting the efficiency analysis.

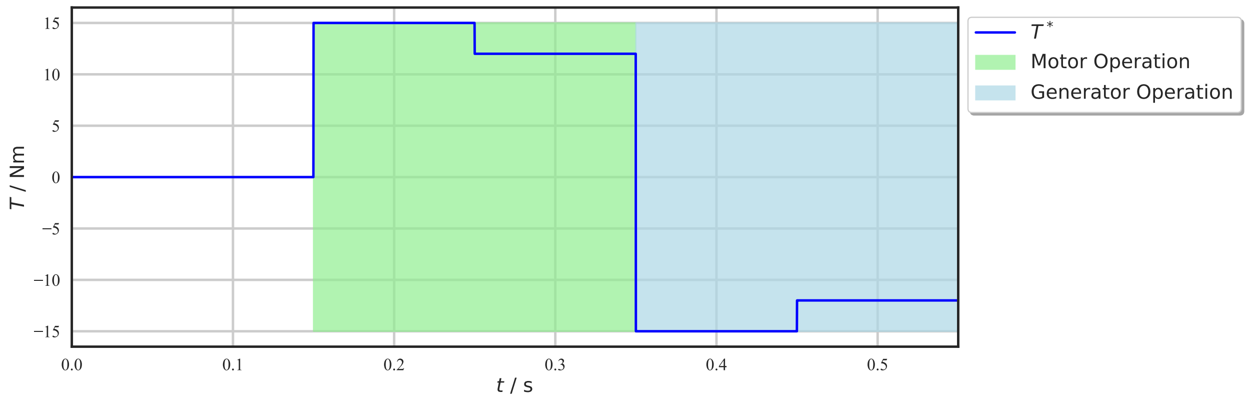


Figure 4-12: Step reference torque input for efficiency analysis at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$

Efficiency of the Agent in Performance Priority Control

The torque tracking performance, already discussed earlier, exhibits the same behavior in figure 4-13. As noted, the excitation current remains at its nominal value throughout the test. While this behavior is advantageous for torque tracking, allowing the agent to manipulate only the stator current due to its faster response compared to the field current, it negatively impacts efficiency. The excitation current contributes significantly to losses, as evidenced in the comparison of stator and rotor losses in figure 4-13. Despite the field current being much lower than the stator current, rotor losses are substantially higher due to the high rotor resistance value listed in table 1.2.

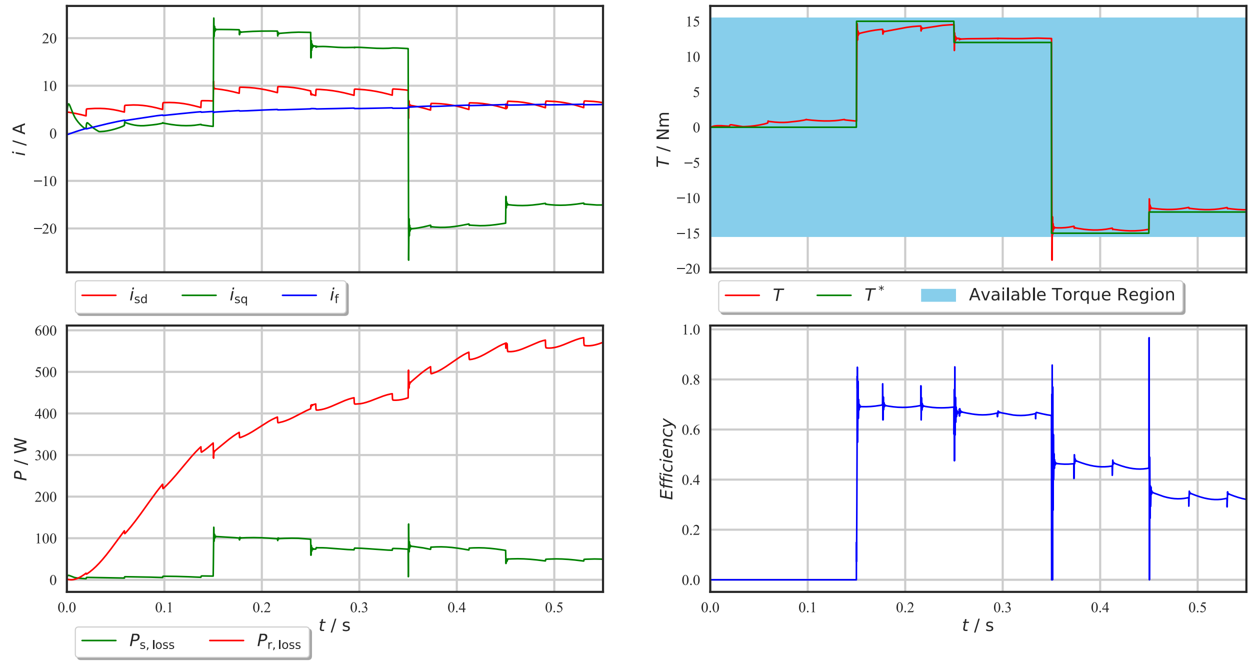


Figure 4-13: Efficiency analysis of performance priority control at the speed of $\omega_{\mathrm{mech}} = 80\mathrm{s}^{-1}$

Then, the available torque region is getting unnecessarily bigger than it is required as the agent applies higher excitation current until it stays at the limit. Consequently, the efficiency is acceptable when all the available torque is used to produce the output power but it is getting poor when the input power is unnecessary high. Therefore, instead of giving the nominal field current for all operations, it is expected to see the excitation current changing with respect to the reference torque in order to improve efficiency in the next section.

Efficiency of the Agent in Efficiency Priority Control

The same torque reference as depicted in figure 4-12 is applied in this environment to evaluate the efficiency of the agent under efficiency priority control. The agent's performance and corresponding efficiency results are presented in figure 4-14.

In the results, the efficiency shows a noticeable increase compared to the previous strategy.

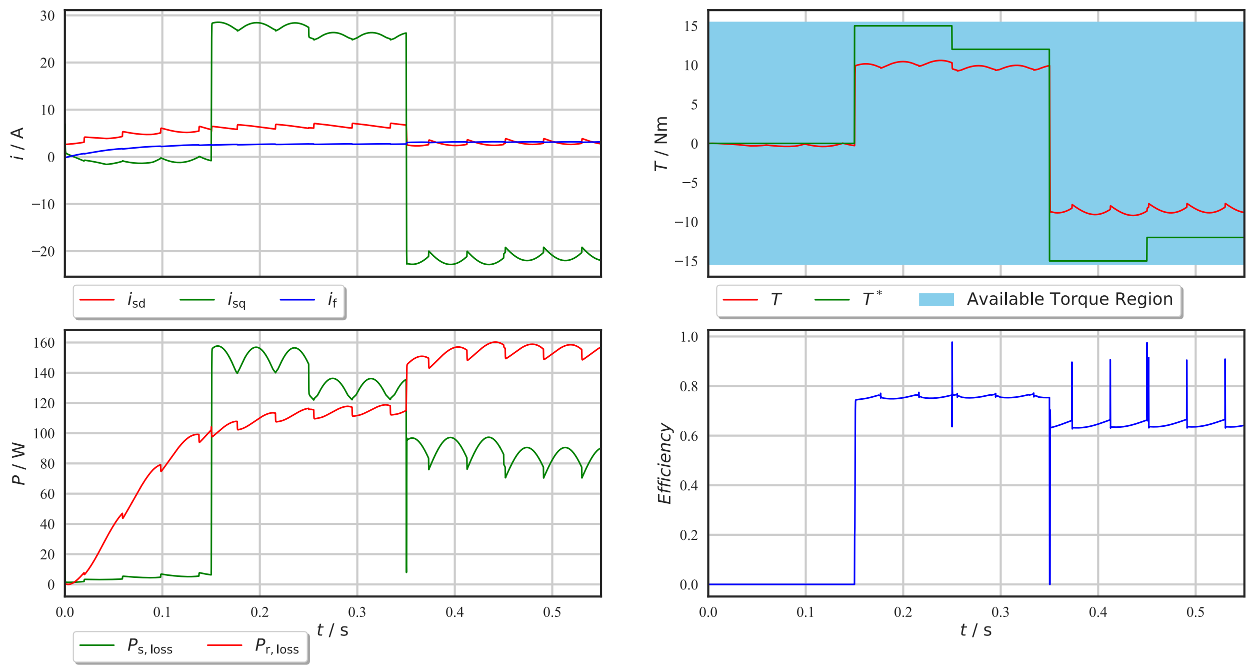


Figure 4-14: Efficiency analysis of efficiency priority control at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$

This improvement is largely due to the agent's strategy of limiting the excitation current, which substantially reduces the rotor losses. In the performance priority control, rotor losses were a dominant factor in the overall losses. By reducing these losses in the efficiency priority control, the overall system efficiency is notably enhanced.

However, as previously discussed, this efficiency gain comes at the cost of performance. The agent, constrained by the limited excitation current it chooses to apply, struggles to track high reference torque values effectively. This trade-off highlights the inherent challenge in balancing performance and efficiency in control strategies.

Efficiency Evaluation Against Optimized Efficiency

Although both control strategies were validated under the same environment, the operating points differ based on the produced electromagnetic torque. Due to these differences, a direct comparison of efficiency between the two strategies is challenging. Therefore, the most effective method for comparing efficiency involves establishing an optimized efficiency curve for each operating point and evaluating how closely each strategy's efficiency approaches this optimization. The behavior of the currents, losses, and corresponding efficiency for the optimized case is illustrated in figure 4-15, which spans the entire torque range at a speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$ using the MTPCL method as discussed in section 1.3.4.

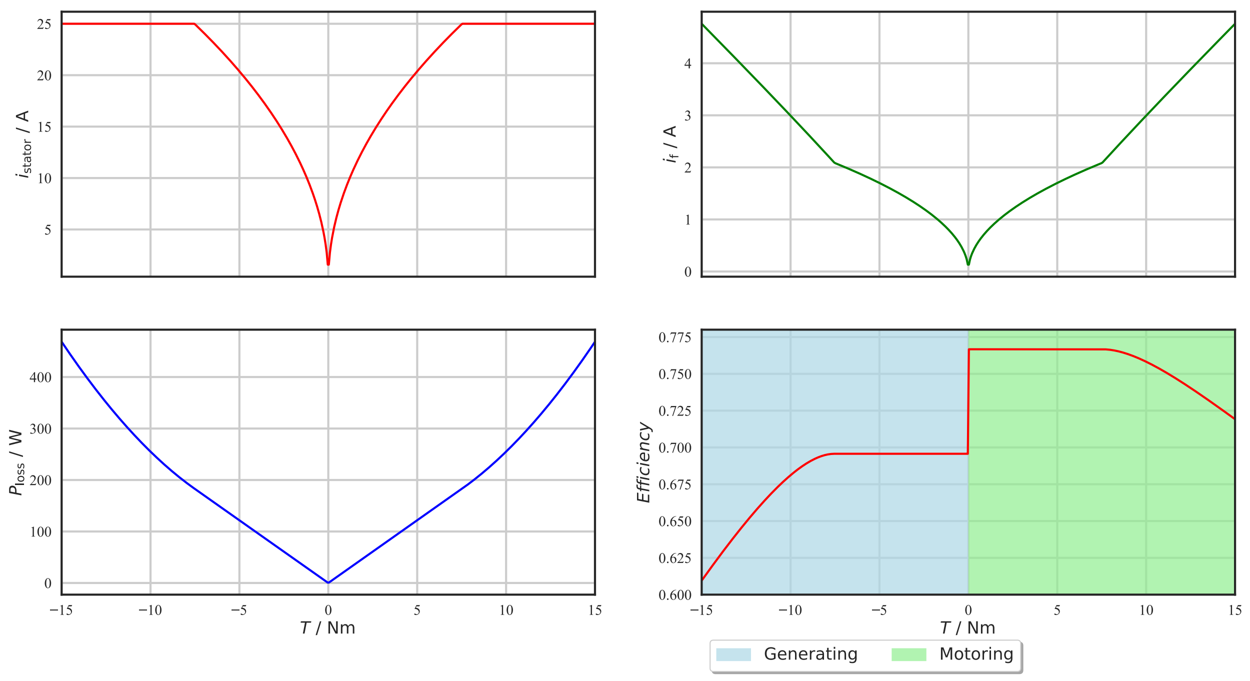


Figure 4-15: Optimization of different operating points at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$

The optimized efficiency, shown in figure 4-15, serves as the benchmark for evaluating the two control strategies. The comparison between the efficiencies of the performance priority and efficiency priority controls against the optimized efficiency across various operating points is depicted in figure 4-16.

As seen in figure 4-16, the efficiencies of both control strategies, along with their respective produced electromagnetic torque are compared to the optimized efficiency for each operating point. The results indicate that the efficiency priority control strategy achieves significantly higher efficiency, particularly during motor operation, where it closely matches the optimized efficiency. In contrast, the performance priority control strategy exhibits notably lower efficiency, especially in the generator operation mode.

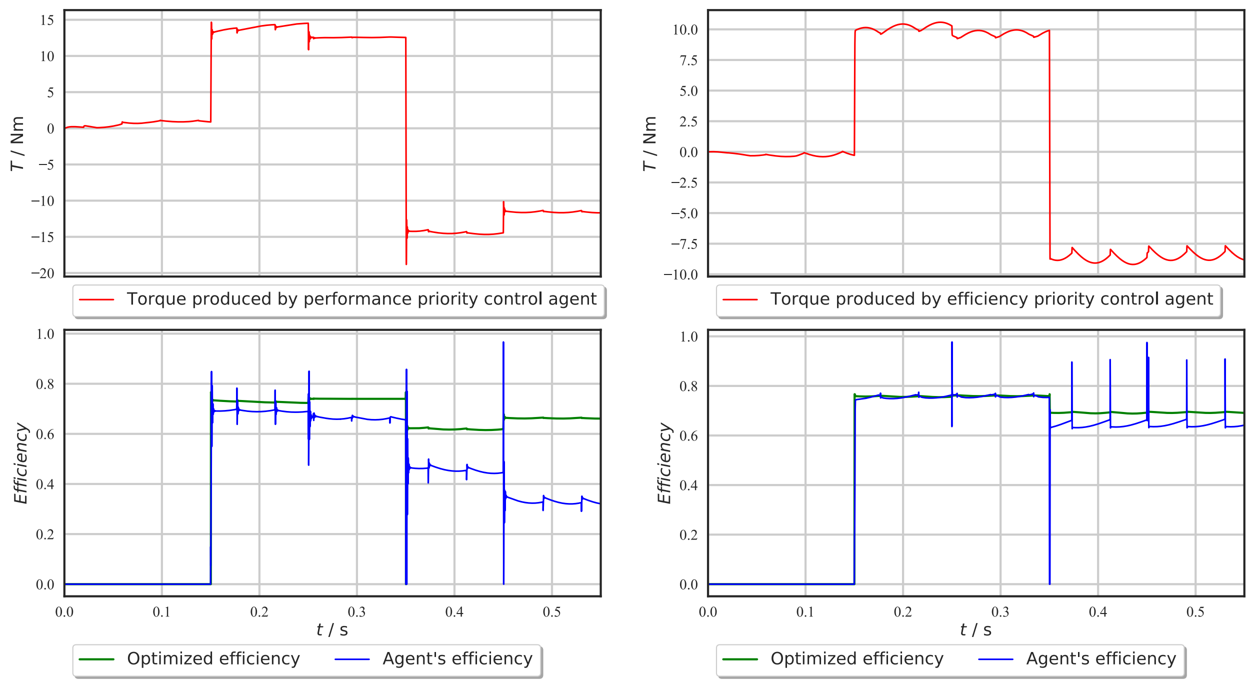


Figure 4-16: Efficiency comparison between two strategies for different operating points at the speed of $\omega_{\text{mech}} = 80\text{s}^{-1}$

These findings suggest that maintaining the excitation current at its nominal value is beneficial for minimizing torque tracking error, but it significantly increases losses, thereby reducing overall efficiency. Conversely, minimizing the excitation current enhances efficiency but makes it more challenging to track high reference torque accurately, particularly due to the slower response of the excitation current.

4.4 Key Takeaways of the Chapter

This chapter demonstrates that optimal torque control of the complex dynamics inherent in an EESM drive can be effectively achieved using RL method. Two distinct strategies were explored: performance priority control and efficiency priority control. Each approach offers unique advantages, making the choice between them contingent on the specific requirements of the application. For applications where precise torque tracking with minimal error is paramount, performance priority control is the preferred strategy. Conversely, if the application allows a broader error tolerance, efficiency priority control proves to be more beneficial. Ultimately, this chapter highlights the challenges of balancing performance and efficiency in RL-based control of EESM, particularly with respect to the management of excitation current.

Chapter 5

Conclusions and Outlook

EESM represents a complex dynamic system within AC drives, involving the interaction of three currents along the d-, q-, and f-axes, with significant coupling between these axes. This thesis has demonstrated that RL can effectively control this intricate motor in a continuous manner.

5.1 Conclusions

This thesis employed a systematic approach, transitioning from CCS current control to CCS torque control, ensuring a smooth progression at each stage. The initial focus on CCS current control demonstrated the RL agent's capability to manage and control the complex dynamics of the EESM drive, successfully tracking reference currents. This step was crucial, as current sensors are typically the primary means of feedback in real-world applications where torque sensors are rarely utilized.

To maintain realism in the control approach, the developed electromagnetic torque was used solely for reward generation during the training phase, not for the control purpose. As a medium step towards optimal torque control, the excitation current was regulated by a PI controller within an idealized environment, adapting the reward design of PMSM from [8]. With constant excitation current, the EESM mimicked PMSM behavior from the controller's perspective, and it was observed that the agent could effectively manage the EESM's system dynamics for torque control through stator current manipulation alone.

Finally, the thesis demonstrated optimal torque control using a fully RL-based controller. Two strategies, performance priority and efficiency priority, were explored, emphasizing the significant impact of excitation current on the controller's performance and efficiency. These findings underscore the importance of balancing performance and efficiency in the use of excitation current for effective control of EESM drives. Then, the key insights from this work are as follows:

- The DDPG algorithm of RL is capable of controlling AC drives in a CCS, while previous research [8] demonstrated that DQNs algorithm is suitable for FCS.
- The RL agent can track the reference torque properly, even without direct inclusion of the produced electromagnetic torque in the observation vector.
- The excitation current is crucial in balancing the controller's performance and efficiency, highlighting its importance in overall system control.

In summary, this thesis has provided an initial proof of concept that the EESM can be effectively controlled using RL. This work lays the foundation for further research and development in this area.

5.2 Future Work

While this thesis has established a strong foundation, there are several areas where further research is necessary to advance towards practical, real-world implementation:

- The current analysis has focused primarily on linear region operation, with the agent trained only up to the base speed. Future work should extend the training to cover all operating points, from negative maximum speed to positive maximum speed, including considerations for the saturation region and field weakening.

- To ensure safe and effective training of torque controllers for EESMs in real-world applications, the development of safety shielding techniques is crucial. Such techniques might include mechanisms to disable power electronic converters from applying risky voltages, thereby ensuring safe and risk-free operation during training and implementation.
- The selection of hyper-parameters in this thesis was carried out through trial and error, as hyper-parameter optimization was beyond the scope of this analysis. For future work, a systematic approach to hyper-parameter optimization should be undertaken prior to training.

By addressing these challenges, future research can move closer to realizing the full potential of RL in controlling EESMs in practical applications.

Appendix A

Validation of RL Agent for CCS Current Control

The agent's performance in CCS current control has already been evaluated in section 2.4, where the validation was conducted under conditions of negative low-speed operation. In this section, the validation of the agent at a positive high speed of $\omega_{\text{mech}} = 200\text{s}^{-1}$ is presented in figure A-1, using the same reference currents as in section 2.4.

As shown in figure A-1, the agent successfully maintains both the stator and rotor voltages within the specified limits while effectively tracking the reference currents, similar to its performance at low speed. However, high-speed operation introduces additional challenges, including the need for higher voltages and increased difficulty in control.

While the tracking behavior of i_{sd} and i_{f} closely mirrors that of the low-speed scenario, the performance of i_{sq} at high speed is noticeably less accurate. The gap between i_{sq} and its reference i_{sq}^* is more pronounced, indicating that the agent struggles more with i_{sq} tracking under these conditions. Meanwhile, the tracking errors for i_{sd} and i_{f} remain comparable to those observed at low speed.

This suggests that while the agent is capable of managing current control at high speeds, its performance is not as robust as it is at low speeds. This discrepancy could be due to several factors: the agent may not have encountered high-speed operations frequently during

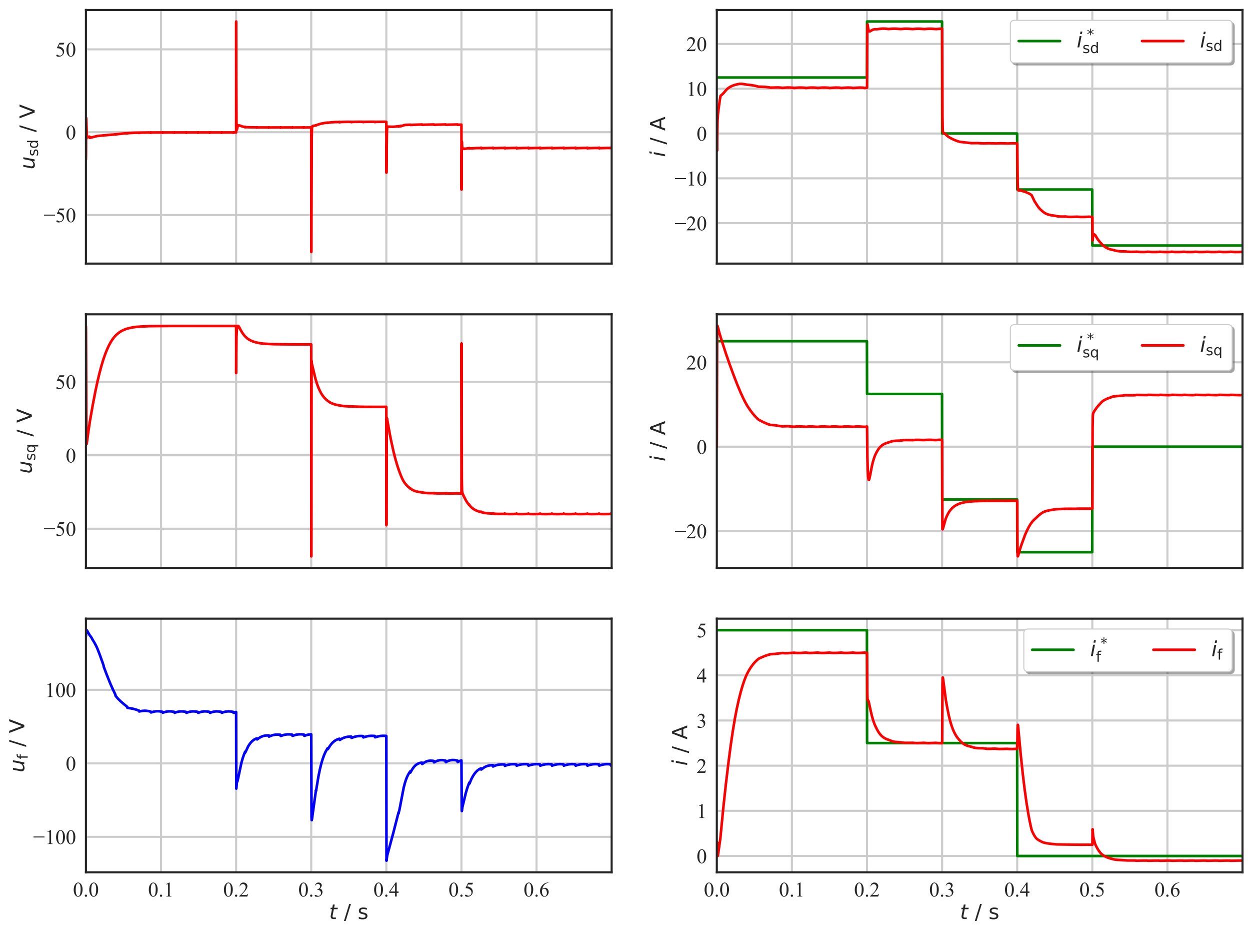


Figure A-1: Validation results at a speed of 200s^{-1} for CCS current control

training, or the increased voltage demands at high speed could have led to more frequent environment resets, resulting in fewer high-speed experiences being stored in the replay buffer.

However, this limitation can be addressed by extending the training duration, allowing the agent to accumulate more experience in high-speed operation, and thereby improving its performance in these scenarios.

Appendix B

Validation of PI-assisted RL Agent

The performance of the PI-assisted RL agent was validated in section 3.4, where validation was conducted under conditions of negative low-speed and negative high-speed operation. In this section, the agent is further validated at a positive high speed of $\omega_{\text{mech}} = 200\text{s}^{-1}$ and a positive low speed of $\omega_{\text{mech}} = 20\text{s}^{-1}$, as shown in figures B-1 and B-2, using the same environment as in section 3.4.

The key difference between positive and negative speed operations is the sign of the stator voltage in the q-axis (u_{sq}): in positive speed operations, the agent applies a positive u_{sq}, whereas in negative speed operations, it applies a negative u_{sq}. However, this difference is not critical as long as the agent successfully tracks the reference torque. Consistent with section 3.4, the applied voltage in high-speed operations is approximately 10 times higher than in low-speed operations, reflecting the speed ratio.

Interestingly, the agent is able to track the high reference torque in both positive high-speed and low-speed operations within the defined tolerance under the nominal excitation current. This suggests that the agent has adequately experienced a wide range of positive speeds during training, while it may have had less exposure to negative high-speed scenarios.

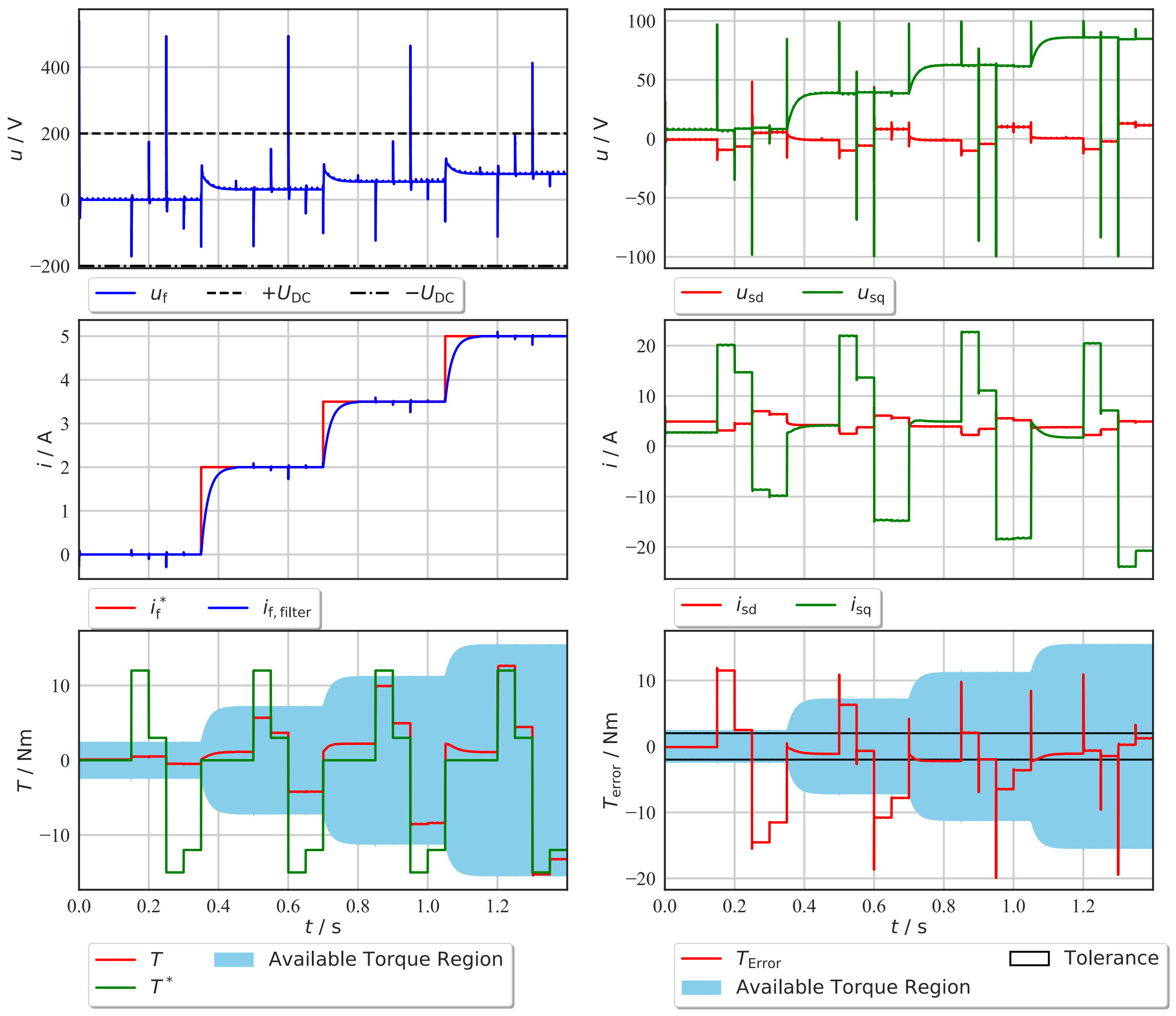


Figure B-1: Validation results at a speed of $\omega_{\text{mech}} = 200\text{s}^{-1}$ for PI-assisted RL control

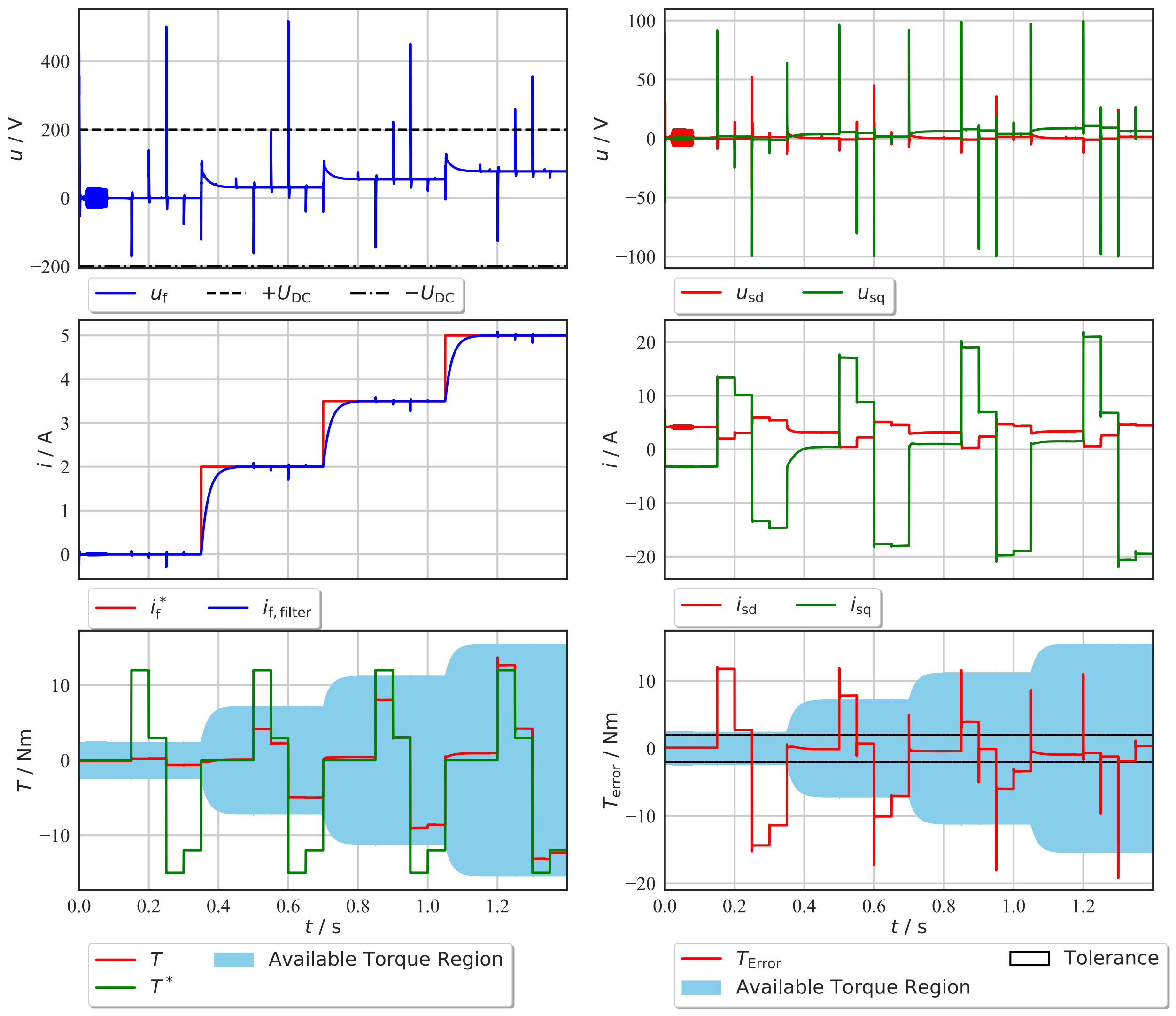


Figure B-2: Validation results at a speed of $\omega_\mathrm{mech} = 20\mathrm{s}^{-1}$ for PI-assisted RL control

Appendix C

Validation of Performance Priority Control Agent

The agent's performance for the performance priority control was previously validated at a positive low speed in section 4.1.3.

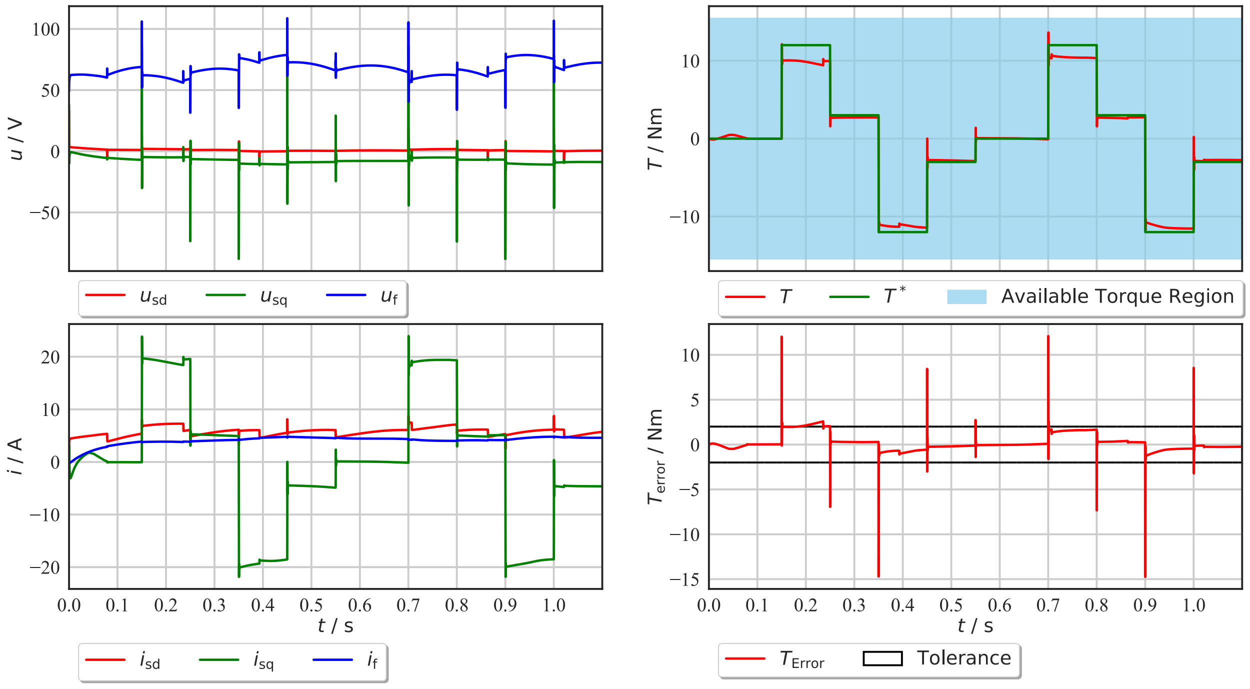


Figure C-1: Validation results at a speed of $\omega_{mech} = -20s^{-1}$ for performance priority control

In this section, the agent is further validated at a negative low speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$ to assess its adaptability across a broader range of operations. The validation is conducted using the same environment as in section 4.1.3.

As shown in figure C-1, the agent's performance remains consistent across all reference torque cycles, regardless of the operating speed. While the torque tracking error is slightly higher than in the positive speed operation, it still falls within the defined tolerance. This indicates that the agent has been effectively trained and can adapt to new environments across a wide range of operating conditions.

Appendix D

Validation of Efficiency Priority Control Agent

The agent's performance for the efficiency priority control was previously validated at a positive low speed in section 4.2.4.

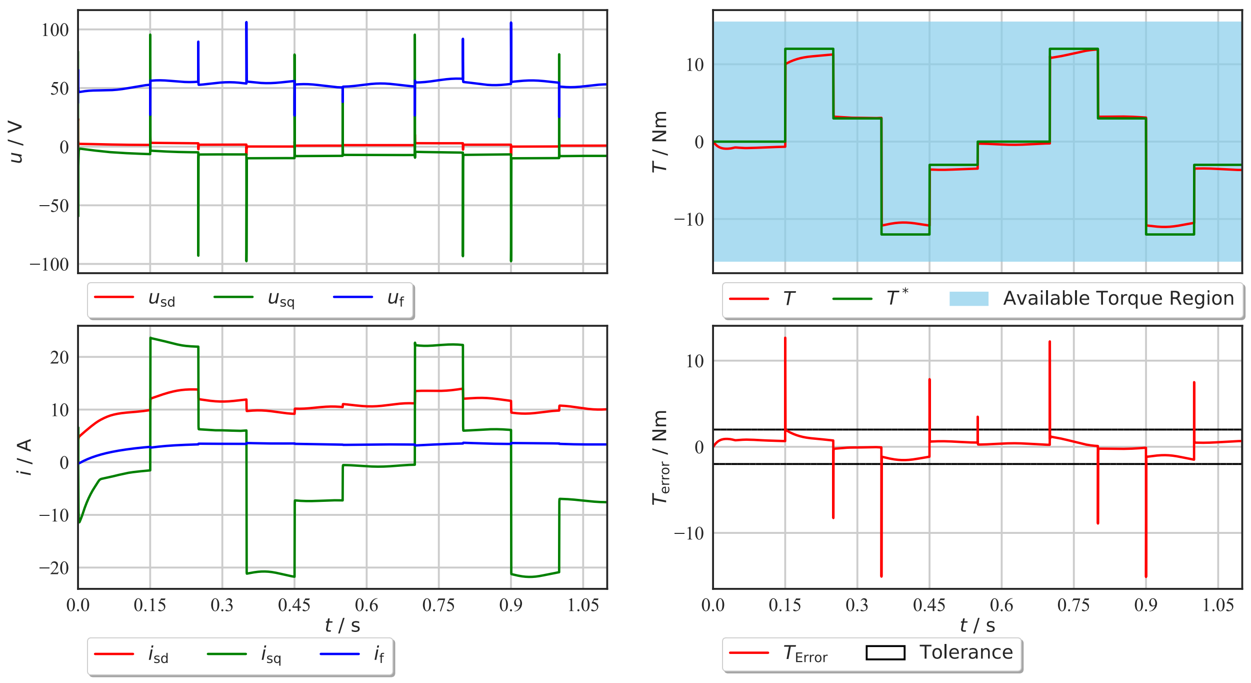


Figure D-1: Validation results at a speed of $\omega_{mech} = -20\text{s}^{-1}$ for efficiency priority control

In this appendix, the agent undergoes further validation at a negative low speed of $\omega_{\mathrm{mech}} = -20\mathrm{s}^{-1}$ to evaluate its adaptability across an even broader range of operational conditions. The same validation environment as described in section 4.2.4 is employed for this assessment.

Figure D-1 illustrates that the agent's performance remains robust throughout all torque cycles, with torque tracking errors nearly identical to those observed in the previous validation. This consistency, regardless of operating speed, demonstrates the agent's effective training and its ability to adapt to diverse operating environments.

Bibliography

[1] M. Koteich, A. Messali, and S. Daurelle in *Self-sensing control of the externally-excited synchronous machine for electric vehicle traction application*, 09 2017.

[2] K. Nikolopoulou, "Easy introduction to reinforcement learning." Scribbr, August 15 2023. Retrieved July 30, 2024, from `https://www.scribbr.com/ai-tools/reinforcement-learning/`.

[3] O. Wallscheid, W. Kirchgässner, M. Schenke, D. Weber, B. Haucke-Korber, D. Jakobeit, and M. Meyer, "UPB LEA Reinforcement Learning Course Materials." `https://github.com/upb-lea/reinforcement_learning_course_materials`.

[4] G. Rösel, N. Daun, A. Giedymin, D. Stojkovic, and M. Töns, "Externally excited synchronous machine (eesm) as main and auxiliary drive," April 2023. 28th of April 2023.

[5] Q. K. Nguyen, J. Schuster, and J. Roth-Stielow, "Energy optimal control of an electrically excited synchronous motor used as traction drive," in *2015 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia)*, pp. 2789–2795, 2015.

[6] B. Podmiljšak, B. Saje, P. Jenuš, T. Tomše, S. Kobe, K. Žužek, and S. Šturm, "The future of permanent-magnet-based electric motors: How will rare earths affect electrification?," *Materials*, vol. 17, no. 4, 2024.

[7] M. Schenke, W. Kirchgässner, and O. Wallscheid, "Controller design for electrical drives by deep reinforcement learning: A proof of concept," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4650–4658, 2020.

[8] M. Schenke and O. Wallscheid, "A deep q-learning direct torque controller for permanent magnet synchronous motors," *IEEE Open Journal of the Industrial Electronics Society*, vol. 2, pp. 388–400, 2021.

[9] M. Schenke, B. Haucke-Korber, and O. Wallscheid, "Safe reinforcement learning direct torque control for continuous control set permanent magnet synchronous motor drives," Aug. 2024.

[10] M. Seilmeier, "Proceedings of the 2011 14th european conference on power electronics and applications," in *Modelling of electrically excited synchronous machine (EESM) considering nonlinear material characteristics and multiple saliencies*, pp. 1–10, 2011.

[11] S. Müller, D. Maier, and N. Parspour, "Inductive electrically excited synchronous machine for electrical vehicles—design, optimization and measurement," *Energies*, vol. 16, no. 4, 2023.

[12] P. Balakrishna, G. Book, W. Kirchgässner, M. Schenke, A. Traue, and O. Wallscheid, "gym-electric-motor (gem): A python toolbox for the simulation of electric drive systems," *Journal of Open Source Software*, vol. 6, no. 58, p. 2498, 2021.

[13] A. Traue, G. Book, W. Kirchgässner, and O. Wallscheid, "Toward a reinforcement learning environment toolbox for intelligent electric motor control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 3, pp. 919–928, 2022.

[14] P. Cunningham, M. Cord, and S. J. Delany, "Supervised learning," in *Machine learning techniques for multimedia: case studies on organization and retrieval*, pp. 21–49, Springer, 2008.

[15] P. Dayan, M. Sahani, and G. Deback, "Unsupervised learning," *The MIT encyclopedia of the cognitive sciences*, pp. 857–859, 1999.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction.* The MIT Press, second ed., 2018.

[17] H. Dong, H. Dong, Z. Ding, S. Zhang, and T. Chang, *Deep Reinforcement Learning.* Springer, 2020.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[20] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," *31st International Conference on Machine Learning, ICML 2014*, vol. 1, 06 2014.

[21] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

[22] M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. d. Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. T. J. Shen, and O. G. Younis, “Gymnasium,” Mar. 2023.

[23] S. Vukosavić, “Digital control of electrical drives,” *Digital Control of Electrical Drives*, pp. 1–353, 01 2007.