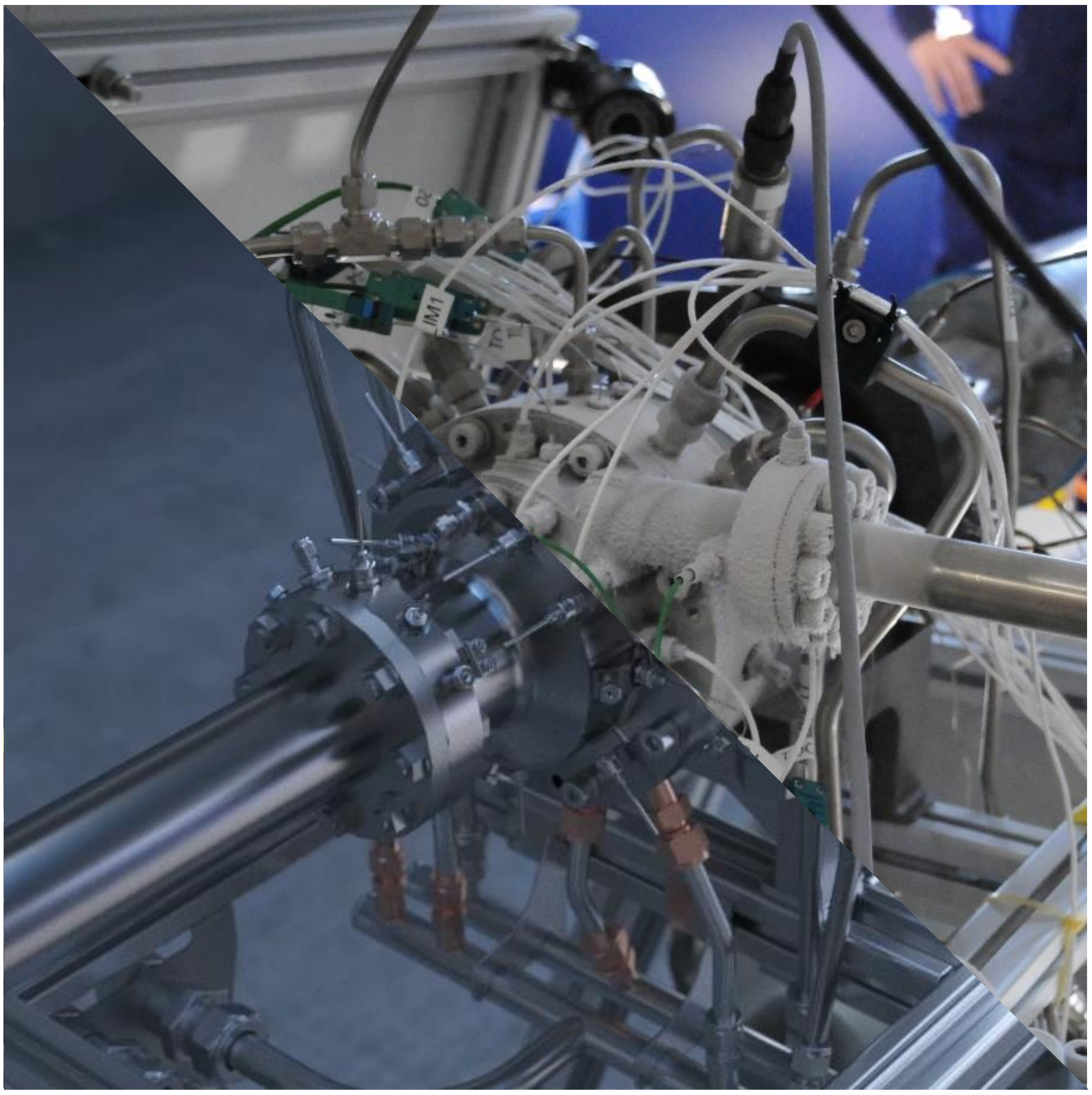


Master’s Thesis

Design and Robustness Analysis of Neural Network Based Control of Rocket Engine Turbopumps

Antonius Adler

August 2022

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

DLR
Deutsches Zentrum
für Luft- und Raumfahrt
German Aerospace Center

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Institute for Computer Science VII
Robotics and Telematics

Master's thesis

Design and Robustness Analysis of Neural Network Based Control of Rocket Engine Turbopumps

Antonius Adler

August 2022

First reviewer:	Prof. Dr. Günther Waxenegger-Wilfing
Second reviewer:	Prof. Dr. Andreas Nüchter
Advisor:	M.Sc. Kai Dresia

Abstract

This master thesis presents the design of a neural network based controller capable of controlling a high-performance fuel turbopump of a liquid rocket engine.
After the giant leaps in rocket engine development in the 1960/70s, only small steps were made to improve the existing technology during the last decades. Now, the NewSpace is an emerging field of new developments to accelerate the commercialization in space and enable challenging space exploration missions to make the first human steps on Mars in future.
Advanced control strategies are a key enabling technology for the next generation of powerful liquid rocket engines designed for reusability. Additional requirements for condition monitoring, multi-restart capability and deep throttling arise from this. Open loop control sequences are no longer sufficient as robust and adaptive control is required to cope with performance degradation and evolving system properties. At the same time, Reinforcement Learning (RL) methods, a branch of machine learning, are improving. Their capabilities have been demonstrated multiple times, for instance, in robotic control or even the control of nuclear fusion reactors, proposing them for rocket engine control applications.
The Deutsches Zentrum für Luft- und Raumfahrt e.V. (German Aerospace Center) (DLR) in Lampoldshausen is currently studying the stationary and transient operation of pump-fed liquid propellant rocket engines within the Liquid Upper-stage deMonstrator ENgine (LUMEN) project for a 25 kN Liquid Oxygen (LOX)/Liquid Natural Gas (LNG) expander-bleed engine. Since this is the first project in Germany that develops and operates a complete rocket engine system, there is a unique opportunity to apply and evaluate modern control strategies in representative rocket engine tests. A major milestone in this endeavour is the demonstration of the LUMEN fuel turbopump control, achieving all control objectives and respecting operational constraints in a preceding test campaign before moving on to the complex control of the entire rocket engine.
The present work comprises the design and robustness analysis of a neural network based controller for this challenging control task. Besides focusing on turbopump specific operational characteristics, an extensive RL framework has been developed to train a controller with RL algorithms in simulation. Considering model uncertainties and other perturbations using curriculum learning and domain randomization, the controller outperforms conservative robust control solutions by adapting to different environments and tasks, thus improving robustness. The qualification for the real-world application in the upcoming test campaign requires an extended robustness analysis, which assessed the effects of individual model parameter variations and combinations using Monte Carlo analyses as well as time-dependent disturbances and performance deterioration. Furthermore, the controller tuning is based on a comprehensive evaluation of individual design decisions valuable for the usage of RL methods for other control applications. The modular and versatile toolset developed can be rapidly adapted to new problems.
Should the controller prove its superior performance to classical and modern control strategies, like Proportional-Integral-Derivative (PID) or Model Predictive Control (MPC) control, new research areas will arise in the field of rocket engine control incorporating machine learning methods.

Kurzfassung

In dieser Masterarbeit wird der Entwurf eines auf einem neuronalen Netz basierenden Reglers für die Regelung einer Turbopumpe eines Flüssigkeitsraketentriebwerks präsentiert. Nach den großen Sprüngen in der Entwicklung von Raketentriebwerken in den 1960/70er Jahren wurden in den letzten Jahrzehnten nur noch kleine Schritte zur Verbesserung der bestehenden Technologie gemacht. Jetzt ist der NewSpace ein aufstrebender Bereich neuer Entwicklungen, die die Kommerzialisierung im Weltraum beschleunigen und anspruchsvolle Raumfahrtmissionen ermöglichen, um in Zukunft die ersten Schritte eines Menschen auf dem Mars zu unternehmen. Fortgeschrittene Regelungsstrategien sind eine Schlüsseltechnologie für die nächste Generation leistungsstarker Raketentriebwerke, die zur Wiederverwendbarkeit ausgelegt sind. Zusätzliche Anforderungen an die Zustandsüberwachung, Mehrfachstartfähigkeit und Schubregelung ergeben sich daraus. Steuerungen reichen nicht mehr aus, da eine robuste und adaptive Regelung erforderlich ist, um Leistungsverluste und sich verändernden Systemeigenschaften zu meistern. Gleichzeitig werden Methoden des selbstverstärkenden Lernens (Reinforcement Learning (RL)), ein Bereich des maschinellen Lernens, immer besser. Ihre Fähigkeiten wurden bereits mehrfach unter Beweis gestellt, zum Beispiel bei der Regelung von Robotern oder sogar von Kernfusionsreaktoren, sodass sie auch für die Regelung von Raketentriebwerken in Frage kommen. Das DLR in Lampoldshausen untersucht derzeit den stationären und transienten Betrieb von pumpengeförderten Flüssigkeitsraketentriebwerkssystemen im Rahmen des LUMEN-Projekts für ein 25 kN LOX/LNG Expander-Bleed Triebwerk. Da dies das erste Projekt in Deutschland ist, das ein komplettes Triebwerkssystem entwickelt und betreibt, bietet sich die einzigartige Gelegenheit, moderne Regelungsstrategien in repräsentativen Raketentriebwerkstests anzuwenden und auszuwerten. Ein wichtiger Meilenstein in diesem Vorhaben ist die Demonstration der Regelung der LUMEN-Treibstoffturbopumpe, in einer vorbereitenden Testkampagne, wobei alle Regelungsziele erreicht und Betriebsbeschränkungen eingehalten werden sollen, bevor die komplexe Regelung des gesamten Raketentriebwerks entwickelt werden kann.

Die vorliegende Arbeit umfasst den Entwurf und die Robustheitsanalyse eines auf einem neuronalen Netz basierenden Reglers für diese anspruchsvolle Regelungsaufgabe. Neben der Fokussierung auf die spezifischen Betriebseigenschaften der Turbopumpe wurde eine umfangreiche Software entwickelt, um einen Regler mit RL-Algorithmen mit einem Simulator zu trainieren. Unter Berücksichtigung von Modellunsicherheiten und anderen Störungen mittels Curriculum Learning und Domain Randomization übertrifft der Regler konservative robuste Regelungslösungen indem er sich an verschiedene Umgebungen und Aufgaben anpasst und so die Robustheit verbessert. Die Qualifizierung für den realen Einsatz in der anstehenden Testkampagne erfordert eine erweiterte Robustheitsanalyse, die die Auswirkungen einzelner Modellparametervariationen und -kombinationen mittels Monte-Carlo-Analysen sowie zeitabhängige Störungen und Leistungsverschlechterung untersucht. Darüber hinaus basiert die Reglereinstellung auf einer umfassenden Bewertung einzelner Entwurfsentscheidungen, die für den Einsatz von RL-Methoden in anderen Anwendungen hilfreich sind. Das entwickelte modulare und vielseitige Softwarepaket kann schnell an neue Problemstellungen angepasst werden.

Sollte sich der Regler gegenüber klassischen und modernen Regelungsstrategien wie PID- oder MPC-Regelung als überlegen erweisen, ergeben sich neue Forschungsfelder im Bereich der Raketentriebwerksregelung unter Einbeziehung von maschinellen Lernmethoden.

- Success in space demands perfection. -

Wernher von Braun

Acknowledgements

My thanks go to all the people who supported me on the way to the Master's degree.
In particular, I would like to thank Prof. Dr. Günther Waxenegger-Wilfing and M.Sc. Kai Dresia for giving me the opportunity to contribute to the research activities at the DLR Institute of Space Propulsion in Lampoldshausen, for their excellent support and for answering almost all my questions. They reinforced my passion for rocket propulsion systems and eagerly shared their experience with me. In the same way, I would like to thank Prof. Dr. Andreas Nüchter for supervising my work.
Last but not least, I want to thank my parents, who advised, encouraged and supported me in all my plans, especially throughout my studies.

Antonius Adler
Würzburg, August 2022

Contents

Acronyms

AI	Artificial Intelligence
BPV	Bypass Valve
CFD	Computational Fluid Dynamics
DA	Domain Adaptation
DDPG	Deep Deterministic Policy Gradient
DLR	Deutsches Zentrum für Luft- und Raumfahrt e.V. (German Aerospace Center)
DR	Domain Randomization
ESA	European Space Agency
ESPSS	European Space Propulsion System Simulation
FCV	Fuel Control Valve
FTP	Fuel Turbopump
GN2	Gaseous Nitrogen
HULC	High performance Universal applicable Lampoldshausen Cluster (Hochperformanter Universell einsetzbarer Lampoldshausen Cluster)
IAE	Integral Absolute Error
IGN	Ignitor
INJ	Injector
ISE	Integral Squared Error
ITAE	Integral Time Absolute Error
JAXA	Japan Aerospace Exploration Agency
LNG	Liquid Natural Gas
LOX	Liquid Oxygen
LUMEN	Liquid Upper-stage deMonstrator ENgine
LSTM	Long Short-Term Memory
MAPE	Mean Absolute Percentage Error
MCC	Main Combustion Chamber

MDP	Markov Decision Process
MFV	Main Fuel Valve
ML	Machine learning
MOV	Main Oxidizer Valve
MPC	Model Predictive Control
NASA	National Aeronautics and Space Administration
NEM	Nozzle Extension Metal
NPSH	Net Positive Suction Head
OCV	Oxidizer Combustion Valve
OP	Operation Point
OTP	Oxidizer Turbopump
PID	Proportional-Integral-Derivative
POMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
RAV	Regenerative Cooling Channel non-Adjustable Valve
RELU	Rectified Linear Unit
RL	Reinforcement Learning
RNN	Recurrent Neural Network
ROF	Mixture Ratio (oxidizer to fuel ratio)
RWRL	Real World Reinforcement Learning
SAC	Soft Actor-Critic
TBV	Turbine Bypass Valve
TCA	Thrust Chamber Assembly
TD3	Twin Delayed DDPG
TDH	Total Dynamic Head
TFV	Turbine Fuel Valve
TOV	Turbine Oxidizer Valve
XCV	Mixer Control Valve

Nomenclature

Liquid Rocket Engines

$\dot{m}$	Mass Flow Rate	[kg/s]
γ	Specific Heat Ratio	
ρ	Fluid Density	[kg/m^3]
A_t	Nozzle Throat Area	[m^2]
c	Effective Exhaust Velocity	[m/s]
c^*	Characterisitic Velocity	[m/s]
F	Thrust	[N]
g_0	Average Sea-level Acceleration of Gravity	[9.806 m/s^2]
H	Pump Head	[m]
h_p	Pressure Head	[m]
h_total	Total Head	[m]
h_v	Velocity Head	[m]
h_z	Elevation Head	[m]
I_{sp}	Specific Impulse	[s]
M	Mean Molecular Mass	[mol]
N	Pump Shaft Speed	[rpm]
N_s	Pump Specific Speed	[rad/s]
P	Pump Power	[W]
p	Pressure	[kg/m^3]
p_e	Nozzle Exit Pressure	[Pa]
p_{cc}	Combustion Chamber Pressure	[Pa]
Q	Volumetric Flow	[m^3/s]
R	Universal Gas Constant	[8.31432 J/K/mol]
T_{cc}	Combustion Chamber Temperature	[K]

v	Velocity	[m/s]
z	Height Difference between Inlet and Outlet	[kg/m^3]
Ca	Cavitation Number	
Cv	Valve Flow Coefficient	
NPSHr	Required Net Positive Suction Head	[m]

Reinforcement Learning

α	Reward Function Configuration Parameter
γ	Discount Factor
π	Policy
ρ_0	Starting State Distribution
τ	Trajectory
θ	Neural Network Parameters
ξ	Domain Parameter
*	Optimal
A	Action Space
a	Action
D	Replay Buffer
H	Entropy
J	Expected Return / Expected Cumulative Reward
o	Observation
P	State-Transition Probability Function
Q	State-Action Value
R	Reward Function
r	Reward
S	State Space
s	State
T	Episode Length
t	Current Time Step
$t-1$	Previous Time Step
V	State Value

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In the new age of spaceflight with increasing commercialization, Europeans are contributing innovative concepts and new solutions to accelerate current developments of future launchers. One main focus is set on improving the efficiency and reusability of rocket propulsion systems to offer independent, less expensive and sustainable access to space. The control and condition monitoring system is crucial for reusable rocket engines, particularly during transient operation phases. In order to meet the increased demands of rocket engines operating at their physical limits, it is necessary to move away from conventional open loop control sequences and instead develop closed loop controllers that also master the classical trade-off between performance and robustness. Furthermore, life-extending, fault-tolerant and robust control are important overriding goals that are key enabling technologies for reusable launch vehicles.[10][11]
The Deutsches Zentrum für Luft- und Raumfahrt e.V. (German Aerospace Center) (DLR) in Lampoldshausen has set itself the goal to overcome traditional ways of thinking and limitations of classical control engineering solutions by using new methods from the field of Machine learning (ML) [12].
Reinforcement Learning (RL)[3] approaches have already demonstrated their power in challenging control tasks, such as plasma control in fusion reactors [13], proposing them as a potential solution for advanced rocket engine control, besides classical PID and MPC [12]. As part of Artificial Intelligence (AI), RL benefits from self-learning through interaction with the environment, normally a simulation model without the need for large datasets or a supervisor. RL algorithms are used to train neural networks capable of finding the optimal control strategy for a given task even in the presence of disturbances while being computationally efficient, real-time capable and less expensive since no extensive testing is needed.[3]
While the performance has already been proven for several rocket engine control problems in simulation [6][14][15][16], the demonstration for real-world applications is still pending. One reason is the required robustness for the transfer from simulation to real-world, as neural network based controllers are prone to model uncertainties and approximation errors. In addition, stability requirements and safety constraints are difficult to verify [17]. Now, the LUMEN demonstrator engine, shown in Figure 1.1, provides a unique opportunity to test advanced control concepts.

The **L**iquid **U**pper stage de**M**onstartor **EN**gine (LUMEN) project was initiated at the DLR in 2017 to gain system competencies in the field of liquid rocket engines and to test new components on system level. These developments became only possible through the continuous improvement of the knowledge about the design, manufacturing and testing of individual subsystems, such as turbopumps, which had been built up in the previous years. The main challenge in this demonstrator project is that, for the first time, different competencies are collaborating and exploring new insights at their interfaces required for the operation of such a complex system. Considering the current developments in the international space propulsion branch, the 25 kN engine is powered by an expander-bleed pump-fed operation cycle supplied with Liquid Oxygen (LOX) as oxidizer and Liquid Natural Gas (LNG) as fuel. The modular breadboard engine offers high modularity and good accessibility, allowing components to be exchanged in a short time so that research institutions and industrial partners can test new designs in a rapid prototyping approach.[18]
A high degree of flexibility is achieved through six electric control valves, allowing a large variety of different system states to be reached. Due to the high complexity, sophisticated control strategies are necessary, for which neural network based control using RL could be one possible solution, as demonstrated by Waxenegger-Wilfing et al.[14] and for LUMEN by Dresia et al.[6]. The LUMEN project will be the first time that the DLR operates a self-developed pump-fed rocket engine and AI-based control can be applied [18].
As major milestone for the verification of modern non-linear control methods towards the LUMEN system level tests in 2023, RL shall be used to control the Fuel Turbopump (FTP) during its test campaign in autumn 2022.

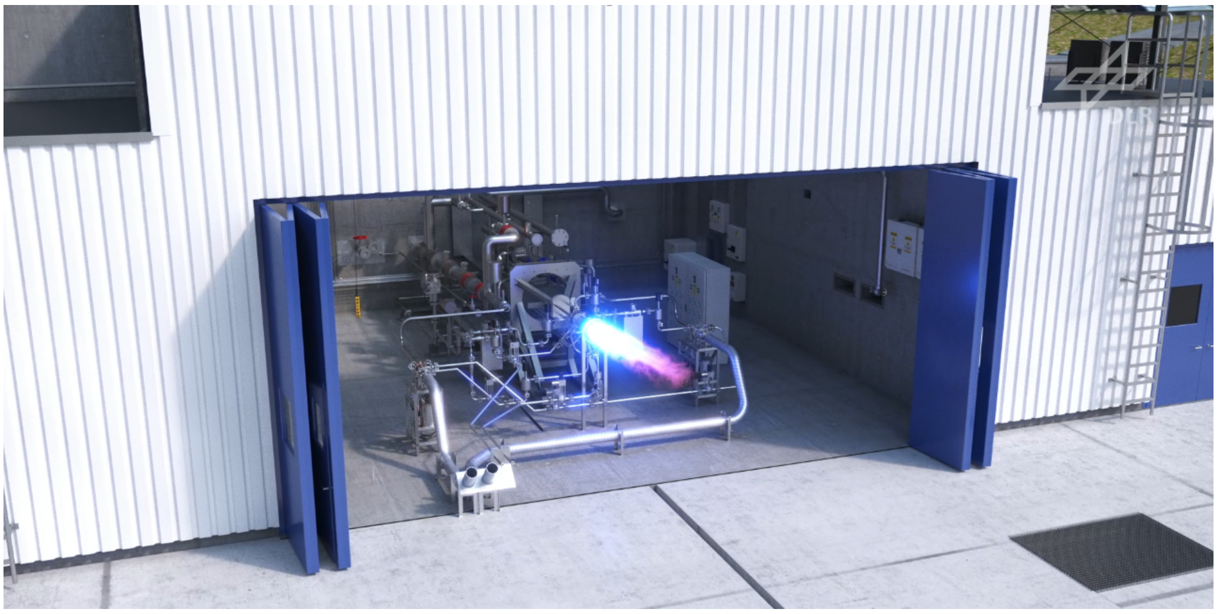


Figure 1.1: LUMEN at Test Bench P8.3 in Lampoldshausen, picture by courtesy of DLR

1.2 Objectives and Approach

In this master thesis, the main objective is the design of a robust neural network based controller for the FTP of the LUMEN demonstrator project based on RL concepts developed by the DLR rocket engine system analysis and control group over the last four years. Moreover, this work also comprises the robustness analysis proving the fundamental transfer capabilities of an RL controller designed in simulation to its application in the real world, also called sim-to-real transfer. This is required for verification prior to the first experimental testing. Challenging control performance requirements resulting from the critical operating conditions of liquid propellant rocket engine turbopumps must be achieved while ensuring that operating constraints are met for damage mitigation and to ensure a safe shutdown.
The theoretical foundations for turbopumps are introduced in Chapter 2, which are crucial for the design and analysis of the controller with regard to the planned real-world application. The introduction to RL in Chapter 3 is followed by an overview of the LUMEN project in Chapter 4. Here, the lectures 'Rocket Propulsion' and 'Deep Reinforcement Learning for Optimal Control' held by Prof. Dr. Günter Waxenegger-Wilfing within the Satellite Technology master's program need to be emphasized as they provided great support for the early introduction.
In Chapter 5, the controller design is presented, starting with the definition of the control task and the required control performance as well as the specification of the operating envelope based on the provided model of the LUMEN FTP and discussions with turbopump experts. On the path to a robust controller design, the RL framework setup and the extensive hyperparameter study in preparation for the final training of the neural network based controller are addressed with a focus on robustness improvements to master the sim-to-real transfer.
The controller analysis in Chapter 6 comprises the analysis of robustness to model uncertainties, noise and delays as well as the analysis of control performance including trajectory tracking and disturbance rejection. After the conclusion of the main results, the next steps towards the LUMEN FTP controller test campaign this year are addressed.
This work comprises the design and robustness analysis of a neural network based controller for the LUMEN FTP using RL. It serves as a proof of concept for future research activities related to real-world applications of RL in the field of liquid rocket engines.

Chapter 2

Fundamentals of Liquid Rocket Engine Turbopumps

This chapter addresses the state of the art of liquid rocket engine turbopumps, starting with a system overview of liquid rocket engines and pump-fed operation cycles followed by the basic principles of turbopumps with a focus on design and operational aspects. It is essential to gain some background knowledge to understand challenges specific to turbopumps and their modeling and the appropriate handling of robustness-enhancing tools for RL. In addition, it will be crucial to analyse the final controller design necessary for a successful sim-to-real transfer later on. Operational requirements, constraints and performance are discussed for the subsequent definition of the turbopump control objectives. Thereby, the reference to the LUMEN project and, in particular, the LUMEN FTP is always kept. The final section presents a brief summary of the state of the art of rocket engine control.
This chapter is based on the standard work for rocket propulsion technology "Modern Engineering for Design of Liquid-Propellant Rocket Engines"[2] unless otherwise stated. This book is highly recommended for further information on this topic.

2.1 Liquid Rocket Engines

From the beginning of space exploration, the rocket engines enabling launchers to reach space were one of the most challenging and exciting pieces of technology people could imagine. They generate the thrust to accelerate the launch system carrying the payload so that the Earth's gravity can be overcome to fly in a specific orbit or to reach another celestial body.
Rocket propulsion technologies are mainly classified based on the propellant type or operating principle. With regard to the LUMEN project, the focus will be set on chemical, liquid propellant rocket engines for upper-stage applications.
To operate a simple pressure-fed bipropellant liquid rocket engine, pressurized tanks, pipes and valves are necessary to supply the combustion chamber with fuel and oxidizer at rated conditions. Control valves are used to ensure engine operation at the required thrust level under optimum combustion conditions. Liquid fuel and oxidizer are injected at the required pressure, temperature and mixture ratio (oxidizer to fuel mass flow ratio) into the combustion chamber,

where they are atomized, evaporated, mixed and finally burned together. In this exothermic reaction, the chemical energy of the reactants is converted into thermal energy, which is converted again into kinetic energy during the expansion of exhaust gases in the nozzle. Based on the conservation of momentum, the rocket is accelerated by the momentum of the exhaust gases ejected in opposite direction.

Liquid rocket engines are most widely used because liquid propellants enable excellent controllability, throttling and restart capability, cooling of structure and produce the highest performance in terms of specific impulse compared to solid propellants and thrust compared to electric propulsion systems. There are also additional challenges such as safe handling, long-term storage, risk of leakage and tank sloshing as well as the additional requirements for cryogenic propellants due to high pressures and large temperature ranges and associated mechanical loads. Further topics that will not be discussed in detail here but are nevertheless crucial for the operation of liquid rocket engines include the ignition process, combustion instability and propellant selection.

The LUMEN engine burns LOX and LNG responding to current developments towards reusability and reducing technical risks as the FTP can be operated at a lower speed compared to liquid hydrogen, enabling the reusage of the Oxidizer Turbopump (OTP) design. While considerable experience with liquid oxygen and hydrogen has been gained from past projects at the DLR in Lampoldshausen, the work with liquid methane is relatively new for rocket propulsion systems. This heritage is mainly because of methane's intermediate performance compared to kerosene and hydrogen as well as the lower density making it less advantageous than kerosene for conventional first rocket stages. With the increasing interest in reusable rocket engines, efforts are intensifying to use methane as a propellant due to its better thermodynamic and combustion properties. The lower mechanical and thermal loads support the lifetime extension for reusability. In addition, LNG is less expensive, which makes it attractive for commercial launchers. In Europe, the Prometheus engine will be the first LOX/LNG rocket engine for future launchers [19].[18][20]

However, until now heat transfer predictions are difficult to obtain for the required supercritical conditions in regenerative cooling channels. Here Waxenegger-Wilfing et al. [21] already showed the power of ML algorithms trained on Computational Fluid Dynamics (CFD) data to generate a surrogate model for regenerative cooling channel design optimization, which was demonstrated by Haemisch et al. [22].

In the following paragraphs, the key performance parameters of a liquid rocket engine are briefly introduced since these are essential for the overall system design of launcher applications. The main concern in launcher design is always the mass budget, in particular the maximization of payload mass while minimizing overall costs, as shown by Dresia et al. [23] in their design optimization of reusable launchers. Here, it should be noted that the LUMEN demonstrator engine is an exception from this in so far as flight application requirements, and space and weight constraints are out of scope [18].

The specific impulse, also called I_{sp}, is the ratio of thrust F per propellant mass flow rate $\dot{m}$ according to Equation 2.1. It is a measure of the rocket engine's efficiency, which is widely accepted for benchmarking launchers and engines.

$$I_{sp} = \frac{F}{\dot{m} g_0} \quad [\mathrm{s}] \tag{2.1}$$

The thrust is a measure of the engine's power (Equation 2.2). The maximum payload mass as well as the achievable acceleration are directly dependent. Besides the thrust produced by the momentum of exhaust gases depending on propellant mass flow $\dot{m}$ and exhaust velocity v_e, there is an additional term for the pressure thrust. The second term accounts for the expansion of the exhaust gases through the nozzle, which causes a pressure difference to the ambient pressure p_a at the nozzle exit (pressure p_e and cross-sectional area A_e at nozzle exit). This contribution becomes significant during the climb phase through the Earth's atmosphere when the ambient pressure decreases exponentially.

$$F = \dot{m} v_e + (p_e - p_a) A_e \quad [\mathrm{N}] \tag{2.2}$$

The exhaust velocity depends on the propellant properties and the combustion conditions depending on mixture ratio as well as combustion chamber pressure (Equation 2.3). As the thrust increases directly proportional to the exhaust velocity v_e, the engine's performance increases with a higher combustion chamber temperature T_{cc}, a lower molecular mass of the propellants M and/or a higher ratio of the pressures at the combustion chamber p_{cc} and nozzle exit p_e (γ specific heat ratio and R universal gas constant).

$$v_e = \sqrt{\frac{2\gamma}{\gamma - 1} \frac{R T_{cc}}{M} \left[1 - \left(\frac{p_e}{p_{cc}} \right)^{\frac{\gamma - 1}{\gamma}} \right]} \quad [\mathrm{m/s}] \tag{2.3}$$

Another important performance parameter of an engine is the effective exhaust velocity c (Equation 2.4), which can be calculated using both previously defined parameters and clearly demonstrates their relationship. Thereby, each parameter expresses a different aspect of performance.

$$c = \frac{F}{\dot{m}} = I_{sp} g_0 \quad [\mathrm{m/s}] \tag{2.4}$$

Finally, the characteristic velocity c^* is an experimentally determinable parameter commonly used for chemical rocket engines. This ratio of the product of combustion chamber pressure p_{cc} and nozzle throat area A_t divided by the mass flow $\dot{m}$ is not a physical quantity but rather a measure of performance to benchmark liquid rocket engine designs.

$$c^* = \frac{p_{cc} A_t}{\dot{m}} \quad [\mathrm{m/s}] \tag{2.5}$$

These performance parameters are important for the rocket engine design and, therefore, essential for turbopump design.[1][2]
Besides the pressure and temperature of fuel and oxidizer at the injection, the propellant mass flow and mixture ratio are significant for the operation of the combustion chamber. The propellant mass flow determines the combustion chamber pressure and thus the thrust level, whereby the mixture ratio is used to control the engine's efficiency and optimal propellant utilization in flight applications. These parameters are also monitored to operate the engine within its limitations derived from the maximum acceptable mechanical and thermal loads. Moreover, additional operating constraints are derived from the engine design, for example tank pressurization, turbopump operation or regenerative cooling determined by the operation cycle.

The LUMEN engine is designed to meet the requirements of the new test bench P8.3 in Lampoldshausen. Besides the test bench interface conditions crucial for turbopump design, the engine scale is limited with a maximal thrust of 25 kN. Responding to the strong competition of new launcher companies, cost reduction and efficiency are becoming increasingly important. As development, manufacturing and operating costs correlate directly with system complexity, LUMEN takes advantage of an expander-bleed operation cycle addressed in the next section.[18]

2.2 Pump-Fed Operation Cycles

Liquid rocket engines are classified by their propellant feed system. Pressure-fed operation cycles pressurize the propellants by the tank pressure, directly feeding them into the combustion chamber via pipes, valves and injector. Since the complexity is reduced because challenging turbopumps are not required, this concept is very reliable and often used for satellite propulsion systems. High thrust levels are not achievable as the combustion chamber pressure is limited by the tank pressure.
More powerful engines use pump-fed operation cycles to maintain a high combustion chamber pressure for a long duration and to reduce the weight of the launcher, mainly by lighter tanks at lower pressure. The core component is the highly complex turbopump, named after the combination of a pump driven by a turbine, which supplies the combustion chamber with liquid propellants at the required pressure and mass flow rate.
Pump-fed operation cycles can be classified into open and closed operation cycles, according to the reuse of the turbine exhaust gas for thrust generation. In closed cycles, after powering the turbines, the fuel- or oxidizer-rich working gas is fed to the combustor to contribute to thrust generation, thus achieving higher efficiency and higher I_{sp}, compared to open cycles that dump the turbine exhaust. In addition, the engine type is specified by the origin of the turbine working gas.
There are five different principal pump-fed operation cycles. The gas generator cycle uses a gas generator and the staged combustion cycle uses preburners to provide the hot gas for the turbines, whereas the expander-bleed and expander cycle benefit from the enthalpy gained by regenerative cooling and the tap-off cycle uses the hot gas from the combustion chamber. The expander cycle and the staged combustion cycle are closed cycles.
Because no preburner or gasgenerator is required and the reduced complexity due to less coupling of the open cycle architecture, the expander-bleed operation cycle was chosen for LUMEN to reduce technical risks and development costs [18]. In addition, this operation cycle is typically used for upper-stage engines and, therefore, representative for the LUMEN demonstrator.
The expander-bleed architecture is simple, reliable and requires less components. A part of the fuel flow is used for combustion chamber wall cooling, whereby the phase changes from liquid to gaseous due to the absorbed thermal energy. The evaporated, heated fuel drives the turbines of the turbopumps to pump fuel and oxidizer into the combustion chamber. This principle is called regenerative cooling, as the enthalpy gained is reintroduced into the operating cycle. Since the turbine exhaust gas is vented, unlike the expander cycle, the total heat energy can be used to drive the turbines, increasing thrust but accepting less efficiency.

The engine power balance is similarly critical as for the expander cycle as the pumped fuel provides its own drive via turbines after regenerative cooling [20]. For this reason, it is one of the key design considerations in engine design and the cause of its inherent susceptibility to production tolerances.

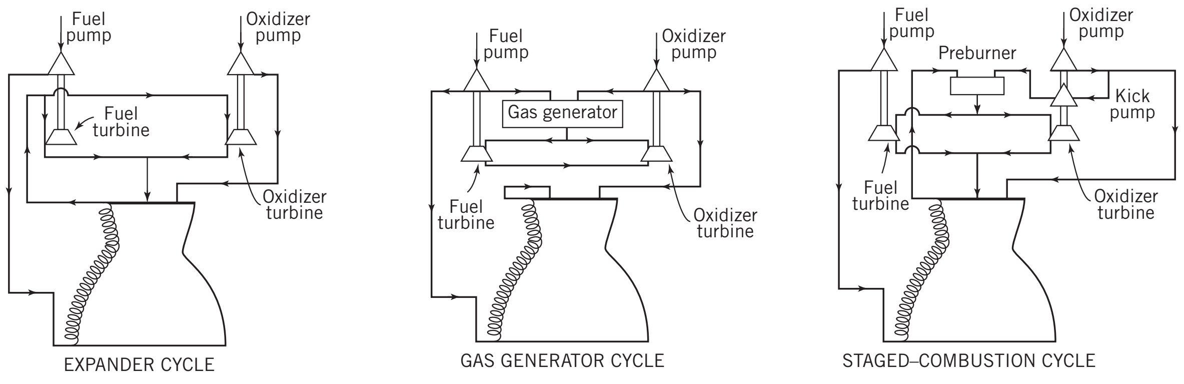


Figure 2.1: Pump-Fed Operation Cycles from Sutton and Biblarz [1, p.243]

Different operating points can be set without much effort, as the turbine mass flow is dumped and thus has no further effect on the system. In addition, the working gas is heated fuel and thus clean and non-corrosive. The main disadvantages of limited chamber pressure due to limited available heat power from regenerative cooling and performance losses due to propellant bleed are irrelevant as efficiency aspects are out of scope for LUMEN. The limitation in combustion chamber pressure allows the design of a safe engine capable of withstanding off-design operating conditions. As the expander-bleed performance is sensitive to component design, particularly the cooling channel design, it is also highly susceptible to production tolerances requiring robust control.[24]

The expander-bleed cycle is used for the LE5-A and LE5-B as well as LE-9 engines developed by Japan Aerospace Exploration Agency (JAXA) and Mitsubishi Heavy Industries Ltd. [25].

Beyond these general considerations, the LUMEN architecture is also remarkable according to Deeken et al. [18] as the parallel turbopump configuration allows different optimizations for fuel and oxidizer supply, regenerative cooling of the nozzle extension provides additional heat power and the mixing of heated and cold fuel allows the injection temperature to be controlled similar to the LE-9 architecture [25]. Besides regenerative cooling, the remixing of heated fuel increases coupling and sensitivity to component design. In addition, LUMEN will be equipped with a total of six electric valves, which have significant speed improvements compared to conventional pneumatic valves, enabling advanced control strategies, such as AI-based control, for transient and steady-state operation.[6]

2.3 Turbopumps

Turbopumps deliver the propellants at the required pressure and flow rate from the tanks to the combustion chamber. The challenging mission requirements are leading to an increasing demand for more powerful engines with reduced tank pressure and higher combustion chamber pressures, which can only be accomplished by turbopumps. Besides high pressure rises and flow rates, the requirements for turbopumps comprise high reliability, high efficiency, low cost and minimum weight as well as throttle-ability. In future deep throttling required for propulsive landing of future reusable launchers will become increasingly important. In order to decrease the propellant tank mass, the turbopump design aims to decrease the pump inlet pressure while ensuring that no cavitation occurs.

The pump design is mainly driven by the required combustion chamber pressure, the engine cycle, the tank pressure and the propellant properties in particular density, viscosity and vapor pressure. In addition, the turbines required to drive the pumps are designed to meet the requirements of the engine cycle, available mass flow and heat energy of the working gas. The combination of turbine and pump also results in additional system requirements for the turbopump. The power transmitting shaft must be able to withstand the extreme temperature differences and to cope with high rotational speeds. Seals, bearings, cooling and lubrication system are essential turbopump components posing additional design challenges. Complex gears prone to wear and tear are somtimes required if two pumps are driven by the same turbine. In general, the selection of cryogenic propellants result in mechanical and thermostructural challenges, which must be considered for material selection and the overall design.[2]

Furthermore, oxidizer pumps require purging systems to separate the turbine side, which is operated with fuel, from the oxidizer pump side, thus avoiding the formation of a flammable mixture. The oxygen pump materials are advanced titanium or steel alloys to cope with the high corrosive environment.[24]

As an alternative approach to conventional turbopumps, the microlauncher company Rocket-Lab from New Zealand designed an electric pump without a turbine for the Rutherford engine and launched it successfully on 25th May 2017 [26]. The reduced system complexity and the high torque for transient operation are advantageous. However, due to the heavy battery packs necessary for its operation, this concept has not been applicable for larger launch vehicles until now.

The development of turbopumps is one of the biggest challenges in propulsion engineering, which the DLR has accepted since the last decade. In the LUMEN project, the recently acquired competencies will be proven for the first time, offering a new perspective for liquid rocket engine development in Germany.

The LUMEN demonstrator is a pioneer for advanced technologies and poses several challenges especially for the turbopumps. The FTP is shown in Figures 2.2 and 2.3. With regard to the expander-bleed architecture, the fuel has to gain enough enthalpy from the regenerative cooling of the combustion chamber and the nozzle extension to drive both turbines. As the turbine exhaust gas is dumped, the turbines are designed for high pressure ratios and small mass flow rates. In addition, the parallel turbopump configuration leads to higher pressure loss over the turbines and smaller mass flow for each turbine resulting in challenging requirements for the turbopump design.[18]

The LUMEN FTP design combines a single-stage centrifugal pump with a supersonic, single-stage impulse turbine. Both subsystems are discussed in the following sections. Due to the high mechanical loads caused by the pressure rise and the temperature gradient between the pump side operated with cryogenic liquids and the turbine side driven by hot gases, the high technological challenge arises besides the material selection in the bearing, cooling and lubrication system. The LUMEN FTPis designed for a nominal pressure rise of 100 bar. For the LUMEN turbopumps, an external lubrication system is used to increase the modularity and controllability of the bearing cooling. The usage of oil instead of the pumped fluid as lubricant supports the reusability, extends the overall lifetime and improves the operation at the test bench.[27][9]

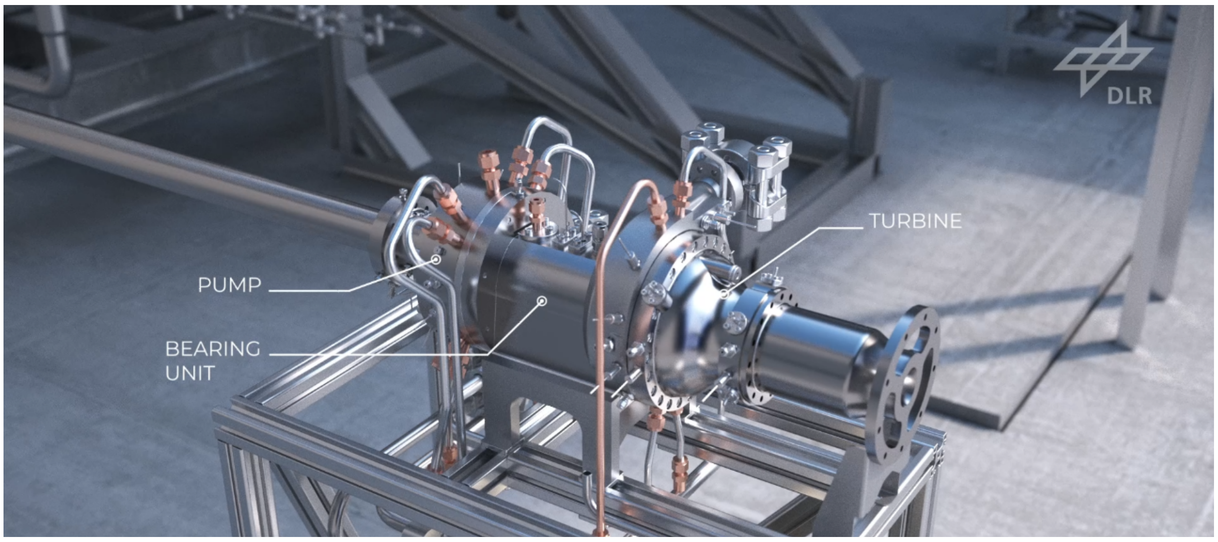


Figure 2.2: LUMEN Fuel Turbopump 3D CAD, picture by courtesy of DLR

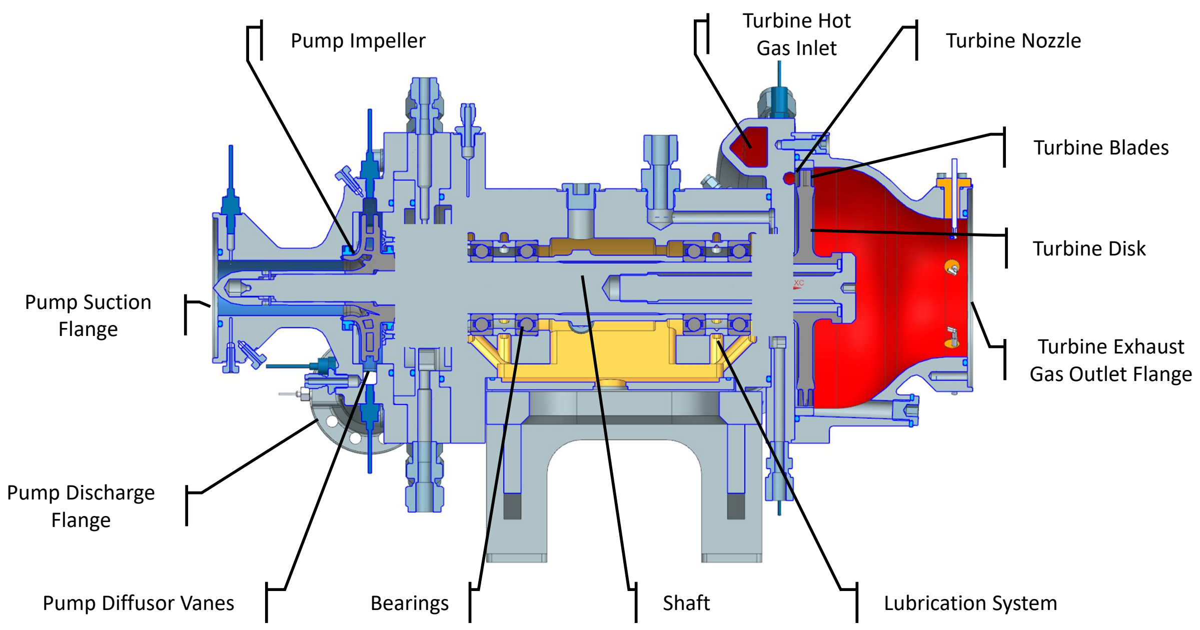


Figure 2.3: LUMEN Fuel Turbopump CAD, picture by courtesy of DLR

2.3.1 Centrifugal Pumps

Pumps employ Bernoulli's principle to increase the fluid's pressure head by adding kinetic energy through the rotor and converting it into static pressure at the stator. The underlying law of energy conservation is described by the sum of pressure head h_p, elevation head h_z and velocity head h_v, which remains constant as long as no losses or gains occur, as stated in Equation 2.6. Gains produced by the pump increase the total energy accordingly. This equation demonstrates that the pump operation is mainly influenced by pressure, volumetric flow and density, which also depends on the fluid temperature.

$$h_{\text{total}} = h_{\text{p}} + h_{\text{z}} + h_{\text{v}} = \frac{p}{\rho g_0} + z + \frac{v}{2g_0} = \text{constant} \tag{2.6}$$

Besides pressure head and volumetric flow, the pump rotational speed and power are important parameters to characterize the pump operation. These are coupled via the following affinity laws of a pump described by Huzel and Huang [2], assuming that pump efficiency is independent of speed for incompressible pumped fluids.

1. Pump volumetric flowrate varies directly with speed

$$Q_1/Q_2 = N_1/N_2 \tag{2.7}$$

2. Pump-developed head varies directly as the square of speed

$$\Delta H_1/\Delta H_2 = N_1^2/N_2^2 \tag{2.8}$$

3. Pump driving power varies directly as a cube of speed

$$P_1/P_2 = N_1^3/N_2^3 \tag{2.9}$$

Considering the technical specification of LUMEN, the mass flow will be used instead of volumetric flow to characterize the pump operation, as an incompressible fluid can be assumed.
The pump specific speed is an important pump design parameter and performance measure for the scaled speed of varying volumetric flow rate and pressure head typical defined for the nominal operation point at maximum efficiency (Equation 2.10). It is derived from the equations stated above. Low specific speeds are characteristic for high pressure head and low flow rate pump designs using radial-type impeller also called centrifugal pump.

$$N_s = \frac{N\sqrt{Q}}{\sqrt[0.75]{g_0 \Delta h_{\text{total}}}} \tag{2.10}$$

Centrifugal Pumps are the most widely used pump type for turbopump applications as these can be operated at high pressures. Axial and mixed flow pump types are used in multistage configurations if high mass flow and less pressure head are required, for example in booster pumps or hydrogen turbopumps.
In Figure 2.3, the LUMEN FTP components are shown. The pump inlet is called the suction side whereas the outlet is the discharge side.

On the suction side, the propellant is supplied from the tank through pipes, fed to the inlet flange and finally to the impeller. The impeller rotates at high speeds to accelerate the fluid through the hydrodynamically formed vanes in radial direction, where it is again decelerated in the diffusor vanes so that static pressure increases. The impeller design aims to reduce axial loads and maximize the fluid velocity ensuring stable flow conditions. After the diffusor, the pump volute directs the fluid to the discharge flange at constant fluid velocity with increasing cross-section to take up the mass flow. In addition, inducers, axial-flow rotors, are needed for pre-pressurization prior to the impeller if otherwise the Net Positive Suction Head (NPSH) required can not be provided and cavitation occurs. In addition, there are the shaft, bearings, seals and the casing with inlet and outlet flanges.

2.3.2 Impulse Turbines

Turbines driving the propellant pumps are powered by converting the working gas enthalpy into the required shaft power. The working gases are supplied under high temperature and high pressure from a preburner, combustion chamber or regenerative cooling system. Impulse turbines are used for high pressure ratios and low mass flow applications, whereas reaction turbines vice versa.
For the LUMEN turbopumps, a single-stage impulse turbine is used where hot gases are expanded in stationary nozzles to convert the heat energy into kinetic energy to maximize its velocity and to align the flow direction before it is applied to the turbine rotor blades. Because of the conservation of momentum, the turbine rotor blades mechanical connected to the shaft are rotating while the working gas is decelerated. The pressure remains constant during the flow through the turbine blades, as the complete enthalpy drop occurs only in the turbine nozzles. In the ideal case for an expander-bleed cycle, the working gas is expanded to ambient pressure, thus achieving the maximum kinetic energy requiring the design of supersonic turbines. The kinetic energy is transferred to the pump side by the turbopump shaft.
As LUMEN is an expander-bleed engine, the turbines are designed for high pressure ratios $\Pi_t = P_{\text{out}}/P_{\text{in}}$ in the order of 13 and low mass flows to cope with the limited available heat energy from regenerative cooling and to minimize the loss of propellant. In addition, LUMEN deploys a parallel turbopump configuration resulting in less mass flow for each turbine which enforces the design of supersonic turbines.[27]

2.3.3 Performance Characterization

In order to compare different turbopumps, taking into account their different designs and operating conditions, system performance parameters have been defined. These parameters are closely related to the performance values already introduced for liquid rocket engines because of their impact on the operation of the entire system. The higher the mass of the turbopump or the tanks and pipes due to increased pump suction pressure requirements, the lower the overall efficiency. The same applies to the turbine mass flow for open cycle types, since the propellant mass required to drive the turbopump does not contribute to thrust generation, thus reduces efficiency in terms of I_{sp}.

Moreover, pump head, turbine pressure ratio, mass flow, rotational speed, efficiency and shaft power are used to characterize the turbopump performance, as it is also common in the hydropower industry.
In pump engineering, performance maps, also called characteristic maps, are used to visualize the operating envelope of a specific pump. In the standard performance maps, pump curves are drawn with volume or mass flow rate on the x-axis and pressure rise or pressure head on the y-axis. The curves are plotted for different speeds or impeller diameters [28]. This results in a chart of characteristic curves in which the relevant parameters for a required operating point are determinable, as depicted in Figure 2.4.

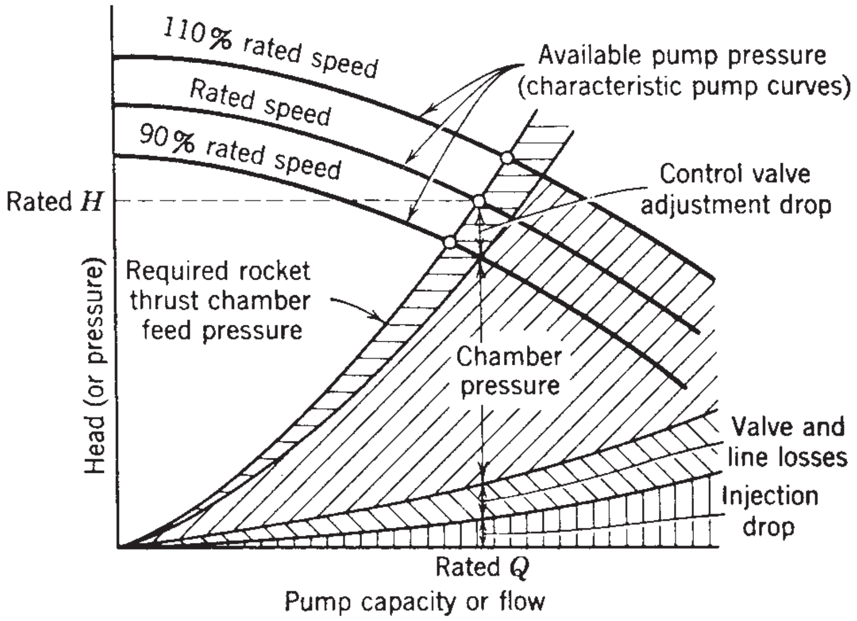


Figure 2.4: Pump Characterisitc Map from Huzel and Huang [2, p.449]

In Figure 2.5, the characteristic map for the LUMEN FTP pump is depicted together with the LUMEN operating points. With the EcosimPro FTP model provided, a grid of operating points is created and the parameters such as rotational speed, pressure rise, mass flow and efficiency are determined. After interpolation, the entire operating envelope can be approximated.
The mass flow rate is plotted on the x-axis, while the pressure rise is shown on the y-axis. The pressure rise is used instead of the pump developed head in order to directly transfer the operation points from the LUMEN technical specification [8] in the pump map. The pump head includes besides pressure rise also friction losses, height differences and fluid velocity (Equation 2.6). The required pressure rise depends not only on the specified injection conditions of the combustion chamber but also on the fluid system pressure losses due to pipes, orifices and valves. Since these losses also vary with pressure and mass flow, the pump curve is always analysed in combination with the system curve to determine the operating point at the intersection of both curves, as shown in Figure 2.4. The coloring of the operational area represents the pump efficiency.

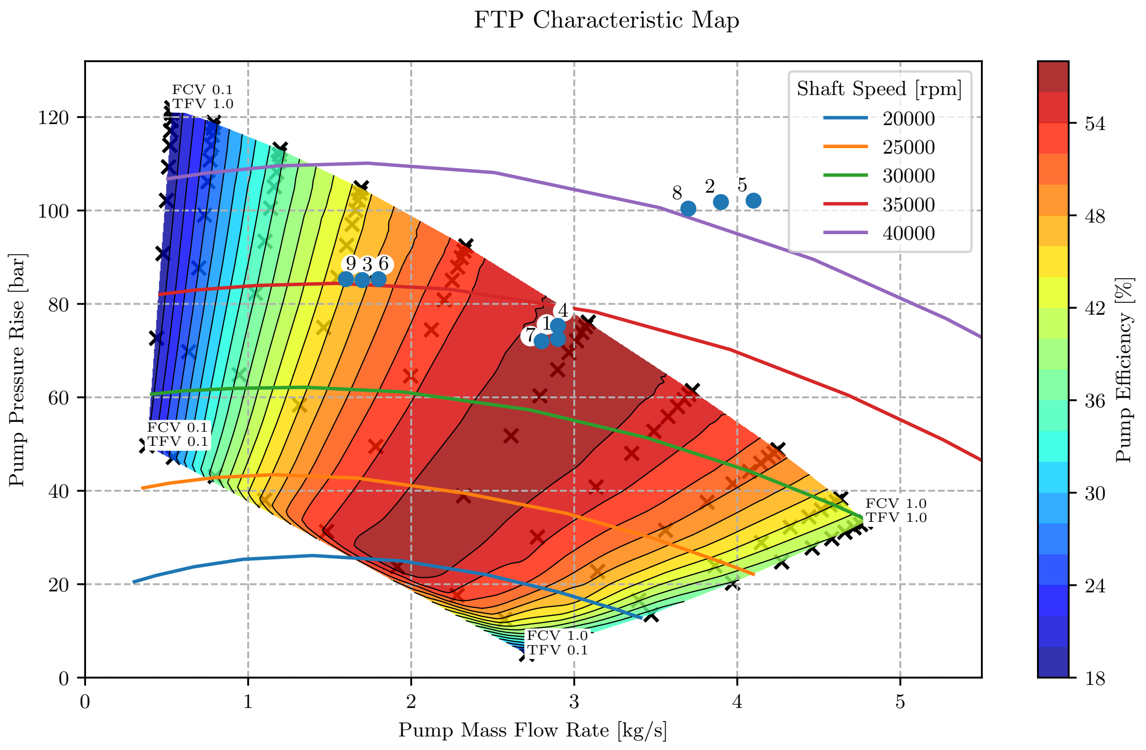


Figure 2.5: LUMEN FTP Pump Characteristic Map

Concerning the test setup (Chapter 4), the Turbine Fuel Valve (TFV) valve on turbine side regulates the available turbine power and the Fuel Control Valve (FCV) valve on pump side controls the pump flow hence the turbopump load. When the TFV valve is opened, the turbine power, rotational speed, pump pressure rise and mass flow rate is increased, and when the FCV valve is opened, the mass flow rate is increased and the pressure rise is reduced. In addition the FCV valve on the pump side is responsible for the achievable efficiency as it directly influences the system curve. Using the pump map, the valve positions can be read for a given operating point. It can also be deduced from the pump map that some operating points lie outside the operating envelope of the turbopump to be tested. This is due to the fact that the turbine side is operated with gaseous nitrogen at standard temperature, thus quite different conditions than in later rocket engine operation when hot gaseous methane is used.
The curves of constant rotational speed are also shown, which are directly related to pressure rise and serve as an indicator of the pump stability. Steep curve slopes will tend to stabilize flow rate and reduce discharge pressure variations. In case that during pump operation the positive slope of the performance curve is reached, control is necessary, otherwise the flow in the pump will stall and the resulting unstable, turbulent flow will lead to extreme loads and vibrations. This effect is called pump surge and limits the pump's throttling capability. For this reason, the closing of the pump flow control valve must be prevented by appropriate safety measures in the controller design. To meet the critical surge requirements when the engine is shut down, the turbine flow control valve should be closed first to shut off the turbine power before the pump flow control valve is closed on the discharge side.[24]

In particular, when considering operational stability during start up, the cavitation condition must be evaluated for the pump. For this purpose, the parameter NPSH has been established. The required NPSH specifies how high the suction pressure head must be to ensure that the acceleration of the fluid in the impeller will not cause the pressure to fall below the vapor pressure, thus reaching the critical point for cavitation. In the case of cavitation, the fluid vapor pressure is exceeded locally, for example at impeller vane trailing edges, creating gas bubbles affecting the fluids compressibility. These lead to flow instabilities resulting in vibrations and performance deterioration. Furthermore, structural damage can be caused by the shock waves generated once the gas bubbles implode. In order to achieve the required NPSH and simultaneously minimize the tank pressure for weight reduction, inducers are used which raise the pressure head already upstream of the impeller. Besides NPSH, the cavitation number can be used as a measure for cavitation and is defined as the ratio of the difference between static pressure and vapor pressure divided by the dynamic pressure of the fluid (Equation 2.12). For Ca ≤ 0, cavitation occurs.
For the LUMEN FTP, an inducer-less design was selected because a supply pressure of at least 7 bar can be ensured to avoid cavitation during test bench operation. In addition, this eliminates a component that is susceptible to wear due to the small distances between the inducer and the pipe wall.[18]

$$\mathrm{NPSH}_r = h_{p_{\text{tank}}} + h_{p_{\text{height/acceleration}}} - h_{p_{\text{losses}}} - h_{p_{\text{vaporpressure}}} \tag{2.11}$$

$$\mathrm{Ca} = \frac{p - p_v}{\frac{1}{2}\rho v^2} \tag{2.12}$$

A major concern during rocket engine development for flight applications is combustion instabilities, besides other instabilities, which can cause Pogo oscillations describing longitudinal launcher vibrations. These are self-excited oscillations in acceleration affecting the propellant in the tank and pipelines, which causes oscillating turbopump suction pressure. Pump suction pressure oscillations and coupled flow oscillations produce thrust oscillations, which are reinforced by feedback on the propellant acceleration and pump inlet pressure oscillations. To prevent catastrophic failures, countermeasures are applied, including the proper design of the resonant frequencies and the use of damping mechanisms. As the propellant is consumed during flight, the launcher mass will decrease and the eigenfrequencies increase. [24]
In general, discharge pressure oscillations and unstable pump flow should be eliminated to avoid combustion instabilities even in the presence of performance degradation or changes in propellant and ambient temperatures.

2.4 Rocket Engine Control

Rocket engine control is required to operate the rocket engine at rated conditions while maintaining all operational constraints. The non-linear control of a liquid rocket engine is one of the most challenging aspects of rocket engine development. Since engine performance and limitations are highly dependent on the rocket engine design, individual controllers must be designed and tuned.

The control objectives include thrust level, mixture ratio and thrust vector control depending on the rocket engine design and the customer mission requirements. Individual control variables and prioritisation are necessary for describing the control tasks and mitigating possible interferences between different control objectives.
Cost reduction and launcher reusability enforce the development of advanced closed loop control methods to cope with evolving engine characteristics and to mitigate thermal and mechanical load transients for lifetime extension. Multi-restart capability and deep-throttling required for the propulsive landing of first rocket stages pose additional challenges. Adjustable flow control valves are used as actuators and flowmeters, pressure and temperature sensors provide measurements as feedback for the controller. Perez-Roca et al.[11] give an overview on rocket engine control and identify several research fields for the control design of future reusable liquid propellant rocket engines. The technical memorandum of Lorenzo et al. [10] on the control for reusable rocket engines has been reviewed with respect to current developments in the field of intelligent control systems, multivariable control, life-extending control and robust rocket engine concept. However, following their survey, no control design has yet been published capable of controlling a wide throttling range.[11]
Until now, open loop control is applied for transient operation, including start, shutdown and set point changes, requiring the development of valve sequences for each engine operation point while maintaining several constraints. Due to production tolerances and changed operation conditions, the open loop control sequences have to be calibrated by adjustment of orifices or valve sequences by intense testing causing high costs. The valve sequences are used in open loop to track set point changes, which are thereafter maintained at near-stationary conditions by a closed loop control routine incorporating the sensor measurements as feedback, for example of the Space Shuttle Main Engine[29]. Other rocket engines like the Vulcain 2 are still operated only with open loop control. Nowadays, the mixture ratio and thrust level control are most important for the steady state operation of a liquid rocket engine defining its operational envelope with thrust and I_{sp}. The mixture ratio control ensures optimal combustion conditions resulting in maximal I_{sp} and maintains the maximal combustion temperature while accounting for secondary objectives such as optimal propellant usage. Because it is critical for safe engine operation, it normally controls the oxidizer mass flow in a fast control loop. The thrust-level control is operated at a lower control frequency to avoid possible interferences. By changing the fuel mass flow, the combustion chamber pressure and thus the final thrust can be regulated while the mixture ratio control ensures optimal combustion. However, this closed loop control approach is only used in near stationary operation. Other rocket engine control applications are start, duration, shutdown or safety control. [11]
In the past, these methods were sufficient since the engine thrust has been rarely changed during the entire flight phase and the engine design and control system have been optimized for the nominal operation. In particular, this was possible because launchers were individually adapted for the orbit of the primary payload. However, current developments attempt to achieve multi-restart capability for launchers to place several payloads in different orbits or deep throttling capabilities for propulsive landing of reusable rocket stages. Moreover, engine specific designed open loop control sequences for transient operation are unable to react to unexpected changed system dynamics or environmental conditions and are very sensitive to disturbances.

In particular, concerning reusability, lifetime shortening peak loads during transient operation should be mitigated with high reliability. The complexity of control systems increases with the number of components, control valves and complexity of the controlled system, such as the coupled system dynamics of pump-fed engine cycle. Advanced control systems are indispensable for this accounting for system requirements, rocket engine dynamics and perturbations.[11]
Using closed loop control enables more stable and precise control and eliminates the need for extensive testing, calibration and tuning of each component. While adaptive control techniques are developed to respond to changing environmental conditions and component characteristics, optimal control aims to maximize the achievable performance for specific operating points. Huzel and Huang describe in the chapter "Control and Condition-Monitoring Systems"[2, p.219 ff.] an expert control system benefiting from a system identification module to modify the adaptive control in the case of unpredictable deviations or disturbances. This approach is similar to the RL-based control even if it does not leverage AI-based strategies.
For closed loop control systems, PI controllers are currently primarily used, which are not capable of optimal control of non-linear systems and taking into account higher-level control objectives such as efficiency maximization. New approaches with MPC are promising but are still very difficult to set up and tune. Perez-Roca et al. [11] give also an overview about advanced control strategies for liquid rocket engines, including PID, MPC and robust control such as H_∞-control. However, in most cases these approaches require linearized models based on several assumptions and the controllers are too specialized and achieve only average performance due to the limited adaptivity[12].
As summarized by Huzel and Huang [2, p.222] for mixture ratio control, the precision of open-loop and partially closed-loop systems is influenced by the following effects, which are also applicable to other control systems:

- instrumentation accuracies
- machining tolerances of orifices
- operating tolerances of regulators
- temperature influences on orifices and regulators
- density tolerance of the propellants, as a function of temperature and of purity
- acceleration effects during flight
- propellant-tank pressure deviations
- turbopump speed deviations
- differences between fuel and oxidizer pump characteristics as a function of speed
- line-resistance changes as a function of temperature and miscellaneous mechanical reasons

For this reason, advanced control strategies with improved performance and robustness are necessary to overcome the limited operational capabilities of current rocket engine control. Moreover, current developments for reusable launch vehicles require innovative control concepts such as life-extending control, fault-tolerant control or deep-throttling. Besides classical PID controller or MPC also AI-based control approaches are under investigation.

At the DLR, new approaches from the field of machine learning are studied for the non-linear control of liquid rocket engines. Waxenegger-Wilfing et al. [14] and Dresia et al. [6] showed the possibilities that RL offers for future rocket engine control. In addition to the offline generation of start sequences [14] and the demonstration of non-linear control of the LUMEN engine in the simulation [6], the first test of a RL-trained controller for a green propellant thruster was accomplished at the test bench M11.5 this year[30]. Intelligent engine control can be the key to more economical and safer engine operation for future launcher applications.

Chapter 3

Reinforcement Learning for Robust Control

Reinforcement Learning (RL) [3] is, besides supervised and unsupervised learning, a part of Machine learning (ML). ML describes a branch of computer science dedicated to the development of algorithms that are able to analyze and derive conclusions about underlying patterns and regularities in datasets. Artificial Intelligence (AI) is the superior term for all methods and techniques that have been developed to imitate human intelligence, for example following biological principles by using neuronal networks and statistical models for decision making.
In contrast to ML approaches that learn from large amounts of data, in RL the agent learns through its interactions with the environment and the returned state and reward of it. In this way, the agent actively influences the next state of its environment and hence the distribution of data it receives, without being dependent on a supervisor or predefined procedures. The agent's main goal is to maximize the cumulative reward through appropriate actions. In training, the agent learns to select initially random actions more and more strategically by exploring the cause-effect relationship between action and received state and reward. Sutton and Barto [3] consider the trial and error approach and the learning about the effects of delayed rewards to be the characteristics of RL. The main challenge lies in mastering general relations that can also be applied to new problems or previously unseen datasets, called generalization. [3]
RL has already been successfully applied in computer games, process control and robotics. In the last few years, the potential application in control engineering has been recognized in order to solve the challenges of classical control theory of the last decades. A special focus is set on robustness improvements of neural network based controller suitable for real-world applications. At DLR the AI and ML working group, led by Prof. Dr. Günther Waxenegger-Wilfing, is dedicated to explore modern approaches for analysis, modeling, control and optimization of liquid rocket engines. Moreover solutions for fault and anomaly detection as well as aspects of reusability and life-extending control of launch vehicles are investigated.[12]
The following chapter introduces the fundamentals of RL and is based on the work of Sutton and Barto [3] unless otherwise declared.

3.1 Idea and Key Concepts

In RL, an agent learns by interacting with its environment from the new state and rewards it receives. The underlying concept is defined by the Markov Decision Process (MDP) applicable for discrete-time dynamical systems.

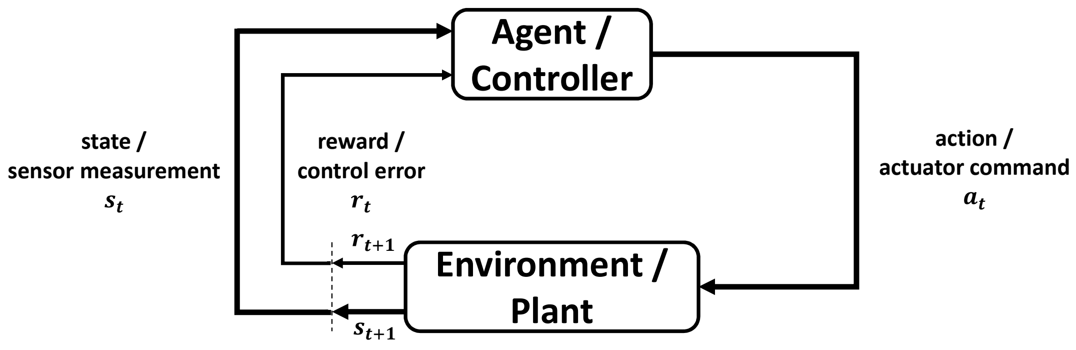

Agent / Controller
state / sensor measurement s_t
reward / control error r_t
action / actuator command a_t
r_{t+1}
Environment / Plant
s_{t+1}

Figure 3.1: Markov Decision Process with Links to Classical Control Theory from Sutton and Barto [3]

The RL agent corresponds to the controller in classical control theory, which is the learner and decision maker. It interacts with the environment which can either exist in the real world or be modeled in simulation.
In the context of robust RL, the environment is also called domain characterized by a set of parameters ξ which are random variables with an unknown probability distribution [31]. New concepts address this by introducing stationary or time-varying uncertainty sets of state-transition probabilities within the framework of robust MDPs [32][33].
Each time step t, the agent's interaction with its environment is described by the action a_t, the resulting new state of the environment s_{t+1}, and the associated reward r_{t+1}. The agent selects its action based on the policy's decision rule. According to the Markov property, the state transition depends only on the most recent state s_t and action a_t. An MDP is defined by the state space S, the action space A, the reward function R, the state-transition probability function P and the starting state distribution ρ_0 (Equation 3.1).

$$\text{MDP} = \{S, A, R, P, \rho_0\} \tag{3.1}$$

The reward function in RL is a measure of the policy's performance which is calculated from the state, the action and the resulting state of the environment, as stated in Equation 3.2. It corresponds to the cost function in an optimization problem.

$$r_{t+1} = R(s_t, a_t, s_{t+1}) \tag{3.2}$$

The state-transition probability function describes the probability $P(s'|s, a)$ of reaching a specific next state when from the current state a certain action is taken.
For control applications, it is important to distinguish between the terms observation and state as the common assumption of full observability is usually not satisfied. The state has to fulfill the Markov property for an unambiguous description of the environment and the effect of an

action on the environment. While the state information can be used in training, when a simulation provides a complete set of state variables, this does not necessarily apply to sensor-based observations in the real world, when the system is only partially observable as non-stationarity or stochasticity is not measurable.[17][34]
The description of the Partially Observable Markov Decision Process (POMDP) introduces environment specific mapping of states to observations $o_t = f_{obs}(s_t)$ [31]. Typical concepts propose the composition of a state by a number of past observations or the usage of Recurrent Neural Network (RNN) to gain information about system dynamics and time derivatives like velocity or acceleration, thus completing the state and fulfilling the assumption of full observability [17][34]. Other approaches introduce belief updates incorporating past observations to improve the knowledge about the true state [33]. This directly relates to the state observer concept in classical control theory including solutions such as the Kalman filter[35]. For simplicity, full observability will be assumed throughout this thesis, as all state information used in training will be available through sensor measurements in later testing.
During training, the agent explores the dynamics and limitations of its environment based on the predefined set of allowed actions and possible states. The ultimate goal is to find the optimal policy to maximize the agent's expected cumulative reward.
The agent's policy links states to probabilities of selecting specific actions (Equation 3.3) and is modeled using neural networks. Neural networks are function approximators to link inputs with outputs and to model the underlying relationships through parametrized statistical models, which are capable to generalize to previously unseen states.

$$a_t \sim \pi(\cdot|s_t) \tag{3.3}$$

To demonstrate these key concepts, the following example of a satellite attitude determination and control system serves as a brief guide. The satellite has a set of actuators such as magnetorquer or reaction wheels and a set of sensors like magnetometer, sun sensor or star tracker. The observed state corresponds to the sensor-based attitude determination and is defined by the attitude, turn rates and moments, assuming full observability. An agent can be trained to control the satellite's attitude based on a reward function considering the difference of the current to the target attitude, for instance pointing to a ground station. Moreover, the optimal policy could find not only the shortest motion but also the economic one, when the energy consumption of the actuators is also included in the reward function.
This example illustrates the reward hypothesis of RL which states "That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward)." Sutton and Barto[3, p.53].
Waxenegger-Wilfing et al.[14, p.2941] summarize the main advantages for control application as follows:

- No derivation of a suitable state-space model, model order reduction or linearization needed
- Direct use of a nonlinear simulation model
- Ideal for highly dynamic situations (no complex online optimization needed)
- Complex reward functions enable complicated goals

According to the authors, the main disadvantage is the lack of verifiability of the stability properties [14], which in principle also goes hand in hand with the challenge of explainability mentioned by Dulac Arnold et al.[17].

3.2 Mathematical Concepts

In order to discuss some algorithms in the next section, the mathematical formalism will be briefly introduced at this point. The agent's performance is evaluated with the cumulative reward to improve the current policy accordingly. For control applications, finite MDPs are analysed where episodes correspond to the length of the reference trajectory to be followed by the agent. In RL, the episode consists of a sequence of states and actions composing the trajectory $\tau = (s_0, a_0, s_1, a_1, s_2, ...)$. For a trajectory with the final time step T, the cumulative reward can be calculated as the finite-horizon undiscounted return $R(\tau)$ as given in Equation 3.4.

$$R(\tau) = \sum_{t=0}^{T-1} r_{t+1} \tag{3.4}$$

A discount factor is introduced in continuing RL problems without a terminal state to ensure a finite return. Given a specific policy, the probability of a certain trajectory $P(\tau|\pi)$ and the respective expected return $J(\pi)$ can be calculated according to Equations 3.5 and 3.6. However, the state-transition probabilities and rewards are unknown in typical RL problems requiring sophisticated learning algorithms.

$$P(\tau|\pi) = \rho_0(s_0)\Pi_{t=0}^{T-1} P(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \tag{3.5}$$

$$J(\pi) = \mathbb{E}_{\tau \sim P(\cdot|\pi)} \{R(\tau)\} \tag{3.6}$$

With regard to the ultimate goal of RL, the search for an optimal policy π^* maximizing the expected return can be formulated with Equation 3.7.

$$\pi^* = \arg\max_{\pi} J(\pi) \tag{3.7}$$

To solve the RL problem, it is necessary to split trajectories to determine the expected reward also for a start from specific states or a specific action from a certain state. Therefore, value functions V for states (Equation 3.8) or action-value functions Q for state-action combinations (Equation 3.9) are defined for a given policy.

$$V^{\pi}(s) = \mathbb{E}_{\tau \sim P(\cdot|\pi)} \{R(\tau)|s_0 = s\} \tag{3.8}$$

$$Q^{\pi}(s, a) = \mathbb{E}_{\tau \sim P(\cdot|\pi)} \{R(\tau)|s_0 = s, a_0 = a\} \tag{3.9}$$

The value function is a measure for the goodness of a state and the Q function for a specific action taken from a given state to maximize the expected return, if the subsequent actions are chosen following the given policy. The advantage function is computed from the difference of action-value and value function to measure how advantageous it is to use a certain action over the average action taken according to a given policy.

The value function can be interpreted as nested functions of itself, proving self-consistency. This relationship is described by the Bellman equations given in Equations 3.10 and 3.11 for an infinite MDP with discount factor γ.

$$V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s),s'\sim P(\cdot|s,a)}\left\{R(s,a,s') + \gamma V^{\pi}(s')\right\} \tag{3.10}$$

$$Q^{\pi}(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}\left\{R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(\cdot|s')}\{Q^{\pi}(s',a')\}\right\} \tag{3.11}$$

In the general RL problem, it is assumed that the system transition dynamics and the reward function are not known to the agent and must be learned during training. With these fundamental relationships, the basic value-based approach to the solution of a RL problem can already be described comprising policy evaluation, policy improvement and policy iteration. In policy evaluation, the value function is computed for a given policy using the Bellman equation. Given the value function, the policy can be improved by acting greedily, hence selecting the action resulting in the next best state. By combining policy evaluation and policy improvement in an iterative process, called policy iteration, the optimal policy can be found. It can be proven that for a finite MDP the optimal policy converges within a finite number of iterations.
Using the Bellman optimality equations, the optimal policy can be found by applying value or Q-value iteration.
Given the optimal policy π^*, the optimal value functions V^*, Q^* as well as the Bellman optimality equations (Equations 3.12 and 3.13) can be derived, which are connected by the following relation Equation 3.14.

$$V^*(s) = \max_a \mathbb{E}_{s'\sim P(\cdot|s,a)}\left\{R(s,a,s') + \gamma V^*(s')\right\} \tag{3.12}$$

$$Q^*(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}\left\{R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right\} \tag{3.13}$$

$$V^*(s) = \max_a Q^*(s,a) \tag{3.14}$$

Based on this mathematical formalismn, dynamic programming can be used to decompose the MDP in subproblems and to reuse solutions to subproblems to solve finally the complete RL problem.

3.3 Reinforcement Learning Algorithms

RL algorithms were developed to solve the MDP. The RL problem can be decomposed into subproblems and solved by the already introduced value functions and Bellman equations. Dynamic programming provides the methodological foundation of RL algorithms, as subproblems occur multiple times and the associated solutions can be reused to find the optimal solution.
Up to now, more than 20 different algorithms have been developed, each having different advantages and disadvantages that qualify them for special use cases. At the beginning, the well-known SARSA and Q-Learning algorithms were used showing the potential of RL even with limited computational resources. Parallel computing on large clusters enabled the solution of more complex RL problems leading to the development of sophisticated algorithms.

Here, the basic concepts are explained so that the special features and differences of selected algorithms can be discussed in the following subsections.
RL algorithms can be categorized in model-free and model-based RL if the value function is learned from experience or a model, knowing the state-transition probability distribution for predictions of future states and rewards. The model can be learned from experience using system identification techniques but will nevertheless be an imperfect approximated representation of the real world. For this reason, the optimal policy can only be as good as the model. In addition, the agent is prone to exploit model specific properties endangering its performance in real-world applications. Models are also used to simulate experiences for model-free algorithms. In this work model-free RL algorithms are used.
Model-free RL algorithms can be distinguished in policy optimization and value-based algorithms. Actor-Critic algorithms are benefitting from both strategies.
Policy optimization uses parameterized policies $\pi_\theta(a|s)$ where the parameters θ correspond to biases and weights of neural networks. Through gradient ascent, the parameters are optimized for a given target function, usually the expected return $J(\pi_\theta)$. These algorithms use only the data generated with the current policy, also called on-policy, offering better convergence properties but requiring significant computational resources due to sample inefficiency. In value-based methods, the Bellman equations are used to determine the optimal Q-function similar to Q-Learning, which will return the optimal action for a specific state, as stated in Equation 3.15.

$$a^*(s) = \max_a Q^*(s, a) \tag{3.15}$$

The Q-function is therefore approximated by neural networks and optimized by gradient descent using a loss function. Since these algorithms, unlike policy optimization, can also use past data from other policies, called off-policy, they are very sample efficient but also more unstable due to increased variance.
Hyperparameter tuning is necessary to configure the RL algorithm properly. Hyperparameters are all variables to parameterize the RL problem prior to training including all variables specific to the RL algorithm.
A major challenge common for all RL algorithms is the trade-off between exploration and exploitation. As already introduced, policy improvement aims to improve the current policy based on value or action-value functions. If the agent always selects the next best action by acting greedily, it will exploit its environment with the risk of overfitting to model specific properties not existing in real world and maybe not even finding the best action. In contrast in exploration the agent selects a random action to explore the entire environment with the risk of underfitting, which means that the agent is unable to learn the underlying relations. In both cases, the agent is not able to cope with unseen situations, thus the optimal balance achieving the best generalization performance is required.
For further reading on RL algorithms, the respective references are recommended. In this section the Deep Deterministic Policy Gradient (DDPG)[36], the Soft Actor-Critic (SAC)[37] and the Proximal Policy Optimization (PPO)[38] algorithms are discussed.

3.3.1 Deep Deterministic Policy Gradient

To learn the optimal action-value function and thus the optimal policy, the Deep Q-Learning algorithm [39] is adapted for the continuous action space with the DDPG algorithm published by Lillicrap et al. [36]. As introduced by the Deep Q-Learning algorithm, experience replay and target networks are used to cope with the correlation of trajectory samples and a non-stationary Q-Learning target [39].
The maximization function of the Q-Learning target is not realizable for continuous action spaces so that an actor-critic architecture is used. The critic corresponds to the Q-function and the actor is an deterministic policy to approximate the maximization operator. The mean squared bellman error is used for off-policy data sampled from a replay buffer to learn the Q-function by gradient descent, whereby at the same time the policy is updated by gradient ascent, assuming that the Q-function is differentiable. In addition, the DDPG algorithm uses Polyak averaging for constant delayed target network updates to stabilize the training. Whereas for discrete action spaces in an ϵ-greedy setting a random action is taken, in continuous action spaces noise is added to the action output for exploration. In DDPG the Ornstein-Uhlenbeck Process [40] is used to generate time-correlated noise.[36]
Further improvements were added by the Twin Delayed DDPG (TD3) algorithm published by Fujimoto et al.[41], which added clipped double Q-learning, delayed policy updates and target policy smoothing.

3.3.2 Soft Actor-Critic

The SAC is an upgraded actor-critic version of DDPG. In particular with regard to real-world applications, the SAC convinces with high sample efficiency using off-policy learning and robustness to hyperparameter configuration, as Tuomas Haarnoja et al. explain in their publications of the SAC [4] and [37].
The SAC is well established for applications in robotics as it needs less samples and hence requires less costly experiments. By using entropy as an additional maximization objective besides the expected cumulative reward it is able to find an acceptable trade-off between exploitation and exploration. Here entropy may be best interpreted as the randomness of a random variable or, more precisely, a measure for the uniformness of a probability distribution and is defined by Equation 3.16.[4]

$$H(P) = \underset{x\sim P}{E}[-\log P(x)] \tag{3.16}$$

The additional entropy regularization prevents the agent from assigning early too high probabilities to specific actions thus enforces more exploration which supports robust and stable training. Both reward components are balanced with the temperature parameter which is adjusted in the course of training. The actor approximates a stochastic policy and uses the Kullbach-Leibler Divergence to ensure that the policy distribution remains close to the Q-function distribution to guarantee that the Q-values are maximized.[4]

```
Algorithm 1 Soft Actor-Critic
Input: $\theta_1, \theta_2, \phi$                                        ▷ Initial parameters
  $\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$                  ▷ Initialize target network weights
  $\mathcal{D} \leftarrow \emptyset$                                      ▷ Initialize an empty replay pool
  for each iteration do
      for each environment step do
          $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$                      ▷ Sample action from the policy
          $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$          ▷ Sample transition from the environment
          $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$   ▷ Store the transition in the replay pool
      end for
      for each gradient step do
          $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$   ▷ Update the Q-function parameters
          $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$                 ▷ Update policy weights
          $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$                   ▷ Adjust temperature
          $\bar{\theta}_i \leftarrow \tau\theta_i + (1 - \tau)\bar{\theta}_i$ for $i \in \{1, 2\}$   ▷ Update target network weights
      end for
  end for
Output: $\theta_1, \theta_2, \phi$                                       ▷ Optimized parameters
```

Figure 3.2: Soft Actor-Critic Algorithm from Haarnoja et al. [4]

3.3.3 Proximal Policy Optimization

The PPO algorithm, in contrast to DDPG and SAC, is an on-policy, policy optimization algorithm and consequently more sample inefficient than off-policy algorithms but convinces with robustness to hyperparameters. It can be used for discrete as well as continuous action spaces. The advantage function used as objective function during gradient ascent is kept close to the previous policy using clipping. The exploration depends on the initial states and the individual runs so that it can be locked in bad local optima.[38]

3.4 Real-World Challenges

When RL is applied to real-world problems, for example in robotics, additional challenges are encountered as assumptions satisfied in simulation are no longer met. Prominent papers by M. Riedmiller [34] and G. Dulac-Arnold, D. Mankowitz and T. Hester [17] have already examined this for control applications and gave a survey of challenges, which are reviewed again in this section with regard to the rocket engine control application.
Riedmiller's intention was to give a hands-on guide for the transformation of "control task specifications into the specification and parameterization of a reinforcement learning task"[34, p.1], thus focusing on the modeling of the RL problem, whereas Dulac-Arnold et al. [17] present meaning, approaches and evaluation of challenges related to RL algorithms.
Given the Markov property, the successor state depends only on the most recent state-action pair, so that the state information must be "rich" enough according to Riedmiller's formulation.

As in real world the sensor data is normally not sufficient to reconstruct all state information, actuator and sensor data of previous time steps are used to deduce information about time dependencies, for example approximated rate of change based on the temporal difference of sensor values. In general, state variables should be reduced to a minimum in order to achieve better generalization properties, faster learning and better control performance.[34]
New research efforts are investigating the use of features that need not to be human-understandable but have such a high information density that the learned agents generalize even better [42].
Regarding the action space, Riedmiller addresses, besides the trade-off with respect to the action set's size in terms of achievable control quality and learning rate, also the requirement that actions have to be sufficient to reach the goal area. This trivial condition can be difficult to prove using the example of the turbopump, when external conditions such as the turbine working gas pressure drops that even completely opening the turbine flow control valve is no longer sufficient to generate the required turbine power.[34]
The reward function should lead to the target region while specifying the overall goals of the RL problem. In the field of rocket engine control, this is a completely new approach since the way how to achieve the control objective is no longer specified, but instead only the overall objective itself, whereby the RL agent has to find the best solution on its own. Comparable to classical control theory, the control frequency must be selected depending on the system dynamics and the required controller performance (see sensitivity and robustness trade-off in subsection 3.6.1).[34]
Riedmiller also discusses other aspects related to the initial state distribution, the terminal state setup or the determination of the episode length that are not relevant to the RL problem considered here. In particular, the initial state distribution can be crucial, but as a well-validated, pretested start sequence will be used for the LUMEN FTP operation, the initial state remains the same. Furthermore, he gives tricks for the normalization of the input values, hint-to-goal heuristics, the selection of the neural network structure as well as exploration and delays. Thereby the experiences at the DLR confirmed that the choice of the number of neurons or layers for a multilayer perceptron is not critical and shows no significant performance differences. System delays are important and have to be considered for observations and actions, which Dulac-Arnold et al.[17] even extend to delayed reward feedback.[34]
In Dulac-Arnold's review of common challenges, sample efficiency, operational safety, real-time capability and partial observability are discussed, relevant for the selection of RL algorithms. The significance of multi-objective reward functions is addressed as a particular challenge, since various goals must be represented by one reward function, where the algorithm may take unintended trade-offs. Based on the derivation that "the global reward function is generally a balance of multiple sub-goals [it follows that] a proper evaluation should explicitly separate the individual components of the reward function to better understand the policy's tradeoffs"[17, p.6]. The explainability of RL policies is also a crucial concern for space applications, since standard qualification and acceptance testing, as well as other verification methods are not applicable. Therefore, innovative safe and fault-tolerant RL concepts are needed to establish acceptance in the space industry[12]. More easily implementable approaches, such as safety layers [43], have already been investigated and are part of current research, as these measures affect the agent's capabilities.

3.5 Sim-to-Real Transfer

In RL, exploration is a key learning method enabling the agent to experience its environment, comparable with trial and error strategies. Testing limitations and learning from failed experiences endangering the agent itself or its environment are essential but hinder training in real-world scenarios, especially for safety-critical applications like rocket engines. In particular with regard to the generation of a large amount of data in a limited time and ensuring safety constraints for unexpected behaviors, the simulation is therefore advantageous for RL environments to train RL agents.[44]
A mathematical model is a representation of a specific part of the real world based on fundamental physical relationships or empirical observations using mathematical concepts like differential equations, stochastic processes or characteristic curves. By describing the behavior of systems in engineering problems, they support the development of a better understanding and enable the analysis without the need for experiments. However, a model is always inaccurate because it is an abstract and simplified representation of the real world and only reflects the properties and behaviors considered during modeling. For this reason, the inherent difference between the simulation and the real world can only be reduced using new scientific knowledge and more computational resources to solve increasingly complex ordinary or partial differential equation systems, but it can not be eliminated. If real test data are available, system identification methods using ML can be used to obtain improved numerical models [12].
Simplified physical relationships, insufficient assumptions, inaccurate calibration and numerical solvers are inevitable approximation errors. This is referred to as the sim-to-real gap or reality-gap. In addition, RL algorithms are prone to exploit the simulator and learn properties that do not exist in the real world. The sim-to-real gap leads to performance degradation, such as lower control accuracy, or worse, a policy that cannot achieve the specified control task. In general, the aim is to reduce the real-to-sim gap through improved system identification and modeling methods.[31]

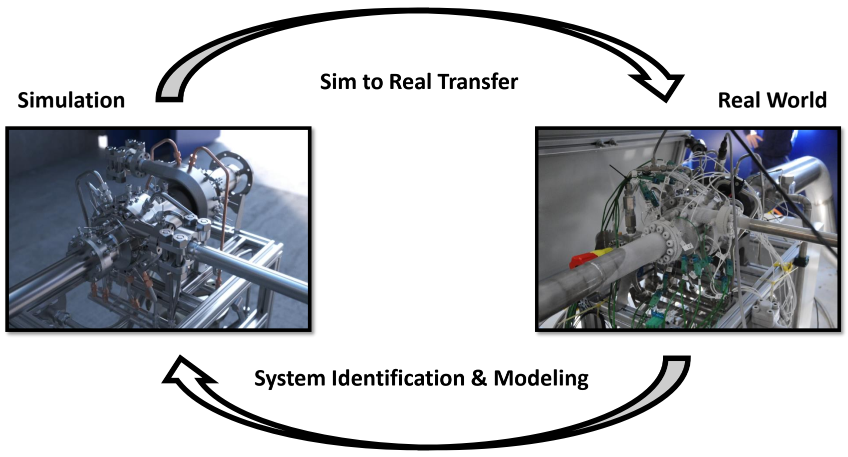


Figure 3.3: Sim-to-Real Transfer Concept, pictures by courtesy of DLR

To overcome the sim-to-real gap during the transfer from simulation (source domain) to reality (target domain), two different approaches are considered. While in zero-shot transfer the agent trained in simulation is directly deployed in the real world, in Domain Adaptation (DA) additional methods are used to prepare the agent in the target domain for its application.[45]
In DA, it is assumed that both source and target domain share common properties that a RL agent's policy learned in simulation can be used to solve the same task in the real world and vice versa [45]. In order to transfer the policy to the target domain, domain knowledge is required, which can be provided by test data, calibration or supervised training in the target domain [46]. The parameters required for calibration are difficult to identify for which feature detection methods from the field of classification are being investigated [42]. Muratore et al.[31] consider these transfer learning approaches as a possible method for sim-to-real transfer, however noted that there were only a few techniques applicable to dynamic systems.
In contrast, the zero-shot transfer is a more straightforward method in which, through accurate modeling or sufficient amounts of simulated experience, the trained agent can be directly deployed in the real world to tackle its tasks. While system identification techniques are used to improve more realistic modeling, Domain Randomization (DR) techniques have become widely established to vary the model during training in such a way that the agent accepts the real world as another variant and adapts to it. [44][46]
ML tries to break up the limitation of system identification, but especially with regard to the robustness of control, DR has gained in acceptance.

3.5.1 Domain Randomization

The research on the sim-to-real transfer of deep neural networks using Domain Randomization (DR) started with the work of Tobin et al. [46] by randomization of visual properties of images for robotic control. The hypothesis of DR states that "if the variability in simulation is significant enough, models trained in simulation will generalize to the real world with no additional training"[46, p.1] and forms the foundation for one of the most effective methods to train neural network based controllers for real-world applications without access to real test data.
In DR, the agent's environment is perturbed, including model parametrization, observations and actions. Through these random variations, DR can be considered as a regularization method that prevents the agent from overfitting to simulator specific features. Therefore the RL goal changes to the maximization of the expected cumulative reward over the set of different environments and thus the distribution of domain parameters, as stated in Equation 3.17.[31]

$$J(\pi) = \mathbb{E}_{\xi \sim P(\cdot)} \left[\mathbb{E}_{\tau \sim P(\cdot|\pi)} \left\{ R(\tau) \right\} \right] \tag{3.17}$$

It is not aimed at achieving a conservative solution where average performance is achieved for the complete set of randomized environments. Instead, the agent should be able to adapt to its environment and respond with appropriate actions to maximize performance for all environments, especially those that are close to the nominal case, similar to adaptive control. That is quite the opposite of conventional robust RL approaches, which try to maximize the reward for the worst-case scenario [47].

For adaption, the structure of the actor is essential to learn the system dynamics and identify deviations online, for example using a history of observations or RNNs [17].
Muratore et al.[31, p.5f.] formulates the following four main design decisions for DR:

- Which parameters should be randomized?
- When should the parameters be randomized?
- How should the parameters be randomized?
- What about physical plausibility?

First, essential parameters for DR must be identified, which, in case of a physical possible deviation from the nominal value, will significantly affect the performance of the RL agent. Model sensitivity analyses are well suited for this purpose. In addition, there are constant or random delays and noise, whose magnitude is best determined by analysis or testing of the real system. The selection of the modifiable domain parameters should always be determined with regard to their validation. In general, the DR parameter set should be reduced to a minimum, as the effort required for their individual evaluation will increase exponentially. Secondly, there are different possibilities when the randomization is applied. Since parameter changes during an episode increase variance and are difficult to implement because simulators would have to be reset, episodic DR has prevailed. Other approaches also include perturbations at random time steps during the episode. Thirdly, the type of randomization determines the type of DR, the training success and the subsequent sim-to-real transfer capabilities. There are different approaches which are explained in more detail in the following subsections. And finally, there is always the question of physical plausibility particularly for complex systems such as turbopumps or complete engines. It is necessary to set up constraints and limitations as well as termination criteria for the simulator depending on the state variables.[31]
This set of design decisions can be extended to questionize which RL algorithm is most suitable for DR as well as how to define a proper termination condition in training.
DR methods are categorized by Muratore et al. [31] in static, adaptive and adversarial domain randomization. It can be argued if other concepts like automatic [48] and active domain randomization [49] need additional categories, as the adaptive domain randomization described by Muratore et al. [31] would require target domain data.
In static DR a fixed distribution of domain parameters is used whereas in adaptive DR the distribution is changed throughout the training. Adversarial DR replaces the probabilistic determination of domain parameters with a second RL problem where an adversary is aiming to worsen the agent's reward by varying domain parameters. Critical hyperparameters for the successful training of the first two methods are the choice of the probability distribution function and how it is being adjusted, and for the last method, the power of the adversarial RL agent.[31]
DR has demonstrated its strength over the last few years. OpenAI demonstrated the learning of robotic hand manipulation with high dexterity despite external disturbances. An Long Short-Term Memory (LSTM) based control policy was trained only in simulation with extensive randomizations and improved modeling of contact sensing. Besides uncorrelated and correlated Gaussian observation noise, normal or lognorm distributed physical model parameters, visual appearance and timing are randomized.[48]

In addition, unmodeled effects are taken into account via randomly chosen action delay and noise as well as randomly applied perturbation forces. Model and action delay randomizations are applied per episode in contrast to timing, perturbation forces and noise which are sampled for each time step. [48]
In the follow-up project, OpenAI applied automatic DR to solve a Rubik's cube using a robotic hand even in the presence of external disturbances. This method introduces a reward-dependent extension of DR boundaries to increase the difficulty level while ensuring continuous learning.[50]
Peng et al. showed that DR can be used to train a control policy for a robot that is able to push different objects to target positions. In addition to the episodic randomization of physical model parameters, the observation noise and the time between two subsequent actions were randomly varied per time step. An LSTM architecture was compared to a feed-forward neural network using an history of past observations and actions to enable system dynamics inference. The evaluation indicated that a feed-forward neural network does not match the performance of an LSTM architecture, but with the help of a history of observations and actions, it improves considerably. As the randomized model parameters are used to train the value function, achieving more useful feedback, lower variance and thus a more stable training for policy gradient algorithms, Peng et al. have referred to this as an omniscient critic. Since trained policies without observation noise and variable action time step achieve significantly worse performance, they deduce that latency and noise are critical DR parameters to enable sim-to-real transfer.[45]
In February 2022, DeepMind demonstrated that RL combined with DR can even be used to control the plasma shape in a tokamak nuclear fusion reactor with unmatched performance and robustness. This is one of the most challenging environments to be controlled due to its high complexity and number of physical and operational constraints. Instead of using a set of PID controllers designed for linearized models in combination with cascade control or target-depending gain scheduling, RL enables the design of non-linear feedback controllers. Efficient simulators based on surrogate models, data-efficient RL algorithm, RNN based critic and fast-to-evaluate actor are the most important features enabling the training on such a complex problem. In addition to the asymmetric learning setup, DR of model parameters, offsets, delays and noise as well as a new method called "learned-region avoidance", preventing the agent by appropriate rewards and termination conditions from entering regions of uncertain dynamics, are used to improve the agent's robustness. Finally, the trained control policy was deployed directly on the tokamak reactor in real-time with a control frequency of 10 kHz, proving its flexibility and generalization capability for high-level control objectives.[13]
As a special form of DR, Active DR proposed by Mehta et al. [49] extends the setup of standard RL problems with a discriminator, which learns a parameter sampling strategy during the training. Focusing on challenging environments can increase the overall efficiency, leading to superior performance and low variance policies compared to standard uniform randomization strategies.[49]

3.5.2 Curriculum Learning and Meta-Learning

Curriculum and Meta-Learning are advanced RL concepts related to the learning process structure. While curriculum learning aims to control training difficulty levels by scheduling different tasks with increasing complexity, meta-learning deals with the learning of different tasks so that

the agent adopts the learning process to transfer it to new tasks.[31]
Curriculum learning is a simple and powerful tool to solve complex problems by leveraging knowledge acquired through simpler tasks in advance. Muratore et al. [31, p.7] state three main challenges which are that the difficulty level is often not determinable, the curriculum learning is not dedicated to finding a robust solution and needs a target distribution which is not given. However, the basic idea can be applied, as OpenAI has demonstrated that with the automatic DR by expanding domain parameter boundaries depending on reaching a certain reward threshold [50].
Although meta-learning, in contrast to DR, aims to solve many different control tasks, similarities can be identified. Thus, task solving in different environments can also be interpreted as different tasks. To adapt to new tasks, meta-learning uses knowledge from past training phases so that RNN or LSTM architectures are needed for the policy network. [31]
Belkhale et al. [51] demonstrated rapid learning capabilities for drones with suspended payloads using meta-learning as a method for adaptive control.
Since no implementation of a recurrent SAC algorithm was currently available in the used Ray Rllib [7] library, this strategy was not applied but should be considered for future applications.

3.5.3 Other Concepts

Robust Markov Decision Processes [32] were introduced to study the uncertainty of state transitions and the robustness of the worst-case combination of model parameter changes [52]. Based on the defintion of robust MDPs Mankowitz et al.[47] developed the so called Robust Maximum A-Posteriori Policy Optimization algorithm that extends the squared temporal difference error of the MPO algorithm [53] by a worst-case value function. The trained policy maximizes the worst-case expected return and omits the problem of learning an average policy for the complete set of possible environments. Mankowitz et al. [47] explicitly discuss the differences to DR and argue that the DR aims at the learning of a policy maximizing the expected return over the whole uncertainty set of domain parameters. In contrast, their proposed algorithm searches for worst-case state transitions, thus enforcing training for the worst-case scenario.
Robust Adversarial Reinforcement Learning relies on the idea that model uncertainties can be handled as external disturbances. Similar to Adversarial DR, an adversarial is trained to minimize the agent's expected reward, thus exploring its weaknesses and enforcing the training on worst-case scenarios. In addition, Pinto et al.[54] proposed the incorporation of domain knowledge, thus providing the opponent more versatility to influence besides the agent's actions also environmental conditions.[54]
Zhang et al. [55] showed that perturbated state observations could be used to increase the robustness of well-known RL algorithms, for instance DDPG or PPO.
Distributional Robustness is another approach to train RL agents with improved robustness to model uncertainties. From a ambiguity set that contains all various models the worst-case setup is taken for training. Other concepts are inspired by adaptive control and use online system identification methods to adapt the policy to the modified environment. Knowledge distillation is a tool to compress complex policy networks. In a teacher-student interaction, the student policy learns the underlying input-output relations from the complex teacher policy while accepting a compromise between less control performance and higher speed due to less complexity.[31]

3.6 Classical Control Theory

The field of classical control theory is so broad that several books can be written on it, such as the reference literature, on which this section is based, from Ogata [56]. This section aims to give an overview of common controller design approaches to compare them with the already introduced RL concepts and to identify potential similarities for later control performance evaluation.

While in RL the definition of control objectives is limited to the formulation of the reward function and thus it is directly specified what needs to be controlled, in classical control engineering different control metrics are used to specify how the system should be controlled, resulting in various, often non-intuitive tuning methods [13]. Generally, the main design goals, stability, robustness, trajectory tracking performance and disturbance rejection, are the same for both approaches.

The closed loop control system structure shown in Figure 3.4 can be compared with the MDP in Figure 3.1. The controller interacts with the system to be controlled, also called plant, using actuators. Their commands are determined by control laws based on the reference input and the feedback measured by sensors. For the consideration of real control tasks, the robustness to model uncertainties , disturbances and noise is also challenging here.

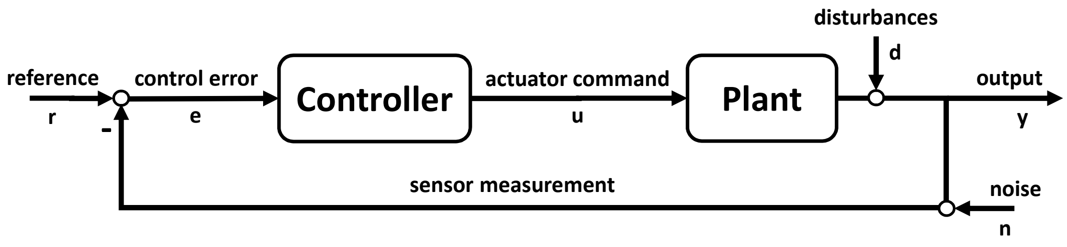


Figure 3.4: Block Diagram of a Closed Loop Control System from Lunze [5]

3.6.1 Sensitivity Trade-off

Considering the classical control loop the trade-off between sensitivity and disturbance rejection becomes evident. Given the transfer functions for the controller $C(s)$ and the plant $P(s)$ in the Laplace domain, the open loop transfer function for the forward path is defined as the product of both in Equation 3.18. The closed loop transfer function considering the feedback loop is stated in Equation 3.19.

$$G_{\mathrm{ol}}(s) = C(s)P(s) \tag{3.18}$$

$$G_{\mathrm{cl}}(s) = \frac{Y(s)}{R(s)} = \frac{G_{\mathrm{ol}}(s)}{1 + G_{\mathrm{ol}}(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)} \tag{3.19}$$

Introducing low-frequency disturbances at the system output $D(s)$, the sensitivity function $S(s)$ can be stated in Equation 3.20. It describes the contribution of the feedback path when comparing $G_{\mathrm{ol}}(s)$ and $G_{\mathrm{cl}}(s)$.

In addition, the complementary sensitivity function $T(s)$, given in Equation 3.21, is used to analyse high-frequency noise on the feedback path corresponding to sensor noise.

$$S(s) = \frac{Y(s)}{D(s)} = \frac{1}{1 + G_{\mathrm{ol}}(s)} = \frac{1}{1 + C(s)P(s)} \tag{3.20}$$

$$T(s) = \frac{Y(s)}{N(s)} = \frac{G_{\mathrm{ol}}(s)}{1 + G_{\mathrm{ol}}(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)} = G_{\mathrm{cl}}(s) \tag{3.21}$$

$$S(s) + T(s) = 1 \tag{3.22}$$

To reduce disturbances and attenuate noise, it is desirable to minimize both terms but as the sum remains constant (Equation 3.22), a trade-off must be chosen. Moreover, as the complementary sensitivity function is equal to the closed loop transfer function, it is also a measure of the control performance to follow a given trajectory. This is often referred to as the limitation of classical control theory. Lunze [5, p.392f.] addresses different design decisions based on this relation. [5]
In normal cases, good trajectory tracking and disturbance rejection properties are required in the low-frequency range, as opposed to noise attenuation for higher frequencies. This can be achieved by an appropriate controller design in the frequency domain. Using bode diagrams, it can be shown that near the cross-over frequency, the sensitivity decreases while the complementary sensitivity increases with increasing frequency. The higher the cross-over frequency, the faster set point changes can be followed and disturbances compensated. However, the sensitivity to high-frequency noise also increases.

3.6.2 Robustness and Stability

Robustness and stability are essential properties of control systems. Unstable plants, disturbances, noise, delays, actuator and sensor limitations are challenging for the controller design regarding stability. Hurwitz criteria, root locus methods or Lyapunov function are well-known tools to determine the stability of control systems but are often based on several assumptions [5]. A stable system that has been displaced from its equilibrium returns to this state again, often referred to as asymptotic stability. In addition, control systems are classified as bounded input and bounded output (BIBO) if the output is always limited for a given input signal range [5]. The definition of robustness can be distinguished into robustness against external disturbances and robustness against model uncertainties. This is analogous to the obstacles during the sim-to-real transfer of RL control policies.

3.6.3 Performance Criteria

Control performance criteria were developed to evaluate specific properties of controlled systems to facilitate tuning and benchmarking of different controllers. Besides metrics to specify the transient response behavior, integral metrics are used to measure the control performance for an entire control trajectory.
To analyse the transient response behavior of the controlled system, test signals with different derivatives, for example impulse, step, ramp and sinus, are applied to the system input and the system output is evaluated with a set of performance criteria, depicted in Figure 3.5.

At the set point change, the time-dependent measures delay, peak time and rise time as well as settling time can be determined. In addition, the peak or overshoot for the difference between the maximum and reference value as well as the steady-state error characterize the control performance. Peaks, overshoots and transients may exceed limitations and drive the engine to a critical operating region that could cause a catastrophic failure. Oscillations can lead to increased wear and tear and cause the system to resonate, making it impossible to keep it under stable control. Rise time and peak time are measures of the control system's ability to follow the reference and how sluggishly it reacts, which must be considered particularly for critical sections of the control trajectory. Settling time is a measure of how long it takes to reach a new set point and stay within a certain tolerance band and is crucial in controller design since the control system is considered to be in a stationary state afterwards.[56]

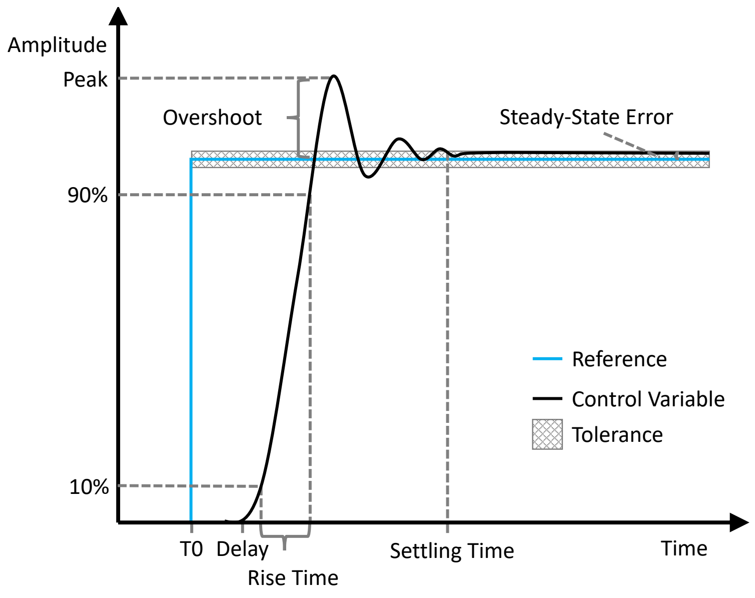


Figure 3.5: Control Performance Metrics

Besides the parameters related to the system dynamics at the set point change, there are so called integral parameters that consider the entire control trajectory. The Integral Absolute Error (IAE) measures the average performance, similar to the Mean Absolute Percentage Error (MAPE).

$$\text{IAE} = \int_{t_0}^{t_0+T} \|e(t)\| \, dt \tag{3.23}$$

The Integral Squared Error (ISE) rates transient values more by squaring so that overshoots, peaks and settling behavior can be assessed.

$$\text{ISE} = \int_{t_0}^{t_0+T} e^2(t)dt \tag{3.24}$$

The Integral Time Absolute Error (ITAE) weights control errors at later time steps more. The transient response behavior is consequently underrepresented due to the multiplication with smaller time quantities at the beginning.

$$\mathrm{ITAE} = \int_{t_0}^{t_0+T} t \,\|e(t)\|\, dt \tag{3.25}$$

Different analysis tools assist in statistical evaluations for design space exploration. Gridding, worst-case and Monte Carlo analysis are methods to structure the robustness analysis over a range of model uncertainties. Gridding examines regularly spaced parameters for specific model uncertainty boundaries. This allows a structured analysis of the entire parametrization space identifying critical regions, which can be investigated with increased resolution using finer grids. Worst-case analyses aim to identify critical combinations of parameter deviations and the resulting worst control performance. For reduced parameter spaces, worst case combinations can be estimated by physical relationships, but with increasing complexity anti-optimization methods gain in importance. In Monte Carlo analysis, a statistical evaluation is performed on thousands of randomly configured samples.The parameters used are chosen according to given probability distributions corresponding to their uncertainty or in the general analysis from an uniform distribution. According to the law of large numbers, the accuracy of the Monte Carlo Analysis improves with an increasing number of samples as the mean of the samples approaches the expected value of the target function. The results of each method allow conclusions about the effect of different model deviations on the control performance and thus on the robustness of the controller. [57]

Chapter 4

LUMEN Project

The **L**iquid **U**pper stage de**M**onstartor **EN**gine LUMEN is dedicated to explore the challenges of rocket engines on system level for the first time at the DLR. The LOX/LNG expander-bleed engine is developed and will be tested at the new experimental test bench P8.3 in Lampoldshausen, shown in Figure 4.1. This 25 kN bread-board engine offers research opportunities on single components and operational concepts up to system tests for research institutions and industrial partners. Therefore, the LUMEN system architecture is highly modular and versatile to increase accessibility for instrumentation and component exchange to maximize the overall test periods while minimizing operational costs. Furthermore, the project objectives of LUMEN comprise the improvement of the DLR competence in rocket engine system analysis in both steady-state and transient cycle analysis by representative operation of a subscale upper stage demonstrator engine. In contrast to the rocket engine development for a specific launcher application, system performance is only a secondary goal and flight requirements as well as size and weight reduction are not considered.[18]

4.1 System Architecture

As the controller design depends on the rocket engine design and system requirements, the LUMEN system architecture has to be discussed. In Figure 4.2, the expander-bleed engine architecture is schematically shown to visualize the fuel and oxidizer flows.
The main drivers for the LUMEN design are cost reduction by limited complexity, modularity, accessibility, lifetime maximization and performance considerations as well as test bench compatibility and representativity for an upper-stage demonstrator engine system. The optimal design is the expander-bleed cycle with a parallel turbopump assembly and partial remixing of heated fuel for fuel injection temperature control. For risk mitigation, two turbopumps are used to optimize their individual operation instead of conventional weight optimized one single shaft LOX/LNG turbopumps.[8][58]
The liquid methane supplied by the test bench interface is pressurized by the LUMEN FTP before one part is fed to the Main Combustion Chamber (MCC) controlled by the Fuel Control Valve (FCV). The remaining cryogenic fuel is used as coolant for combustion chamber wall cooling in a counter-flow arrangement.

To maintain the required injection temperature for supercritical gaseous fuel at injection, the heated fuel is partially mixed with the cold fuel controlled by the Mixer Control Valve (XCV), similar to the Japanese LE-5B expander-bleed engine. XCV is also used to control the cooling channel pressure to maintain supercritical conditions for methane. The remaining heated fuel is used to cool the nozzle extension to gain additional heat energy needed to drive both turbopumps thereafter. Thereby the turbine power is controlled by the Turbine Fuel Valve (TFV) for the Fuel Turbopump (FTP) and the Turbine Oxidizer Valve (TOV) for the Oxidizer Turbopump (OTP). The LOX supplied by the test bench is directly pumped by the OTP into the injector, controlled by the Oxidizer Combustion Valve (OCV). For safe operation, an additional Turbine Bypass Valve (TBV) can be used to release the heated fuel immediately to decelerate both turbopumps. Moreover, the Bypass Valve (BPV) allows the control of the MCC cooling channel mass flow even when the additional fuel mass flow is not used for injection or the turbines. The Main Fuel Valve (MFV) and Main Oxidizer Valve (MOV) are used to start and shut off the combustion as required and a torch igniter mounted at the injector head is used for ignition.[8][18]
Compared to a flight model, this engine is highly controllable, due to its amount of six electric valves instead of orifices and conventional sluggish pneumatic valves, so that the desired operating point can be set even if operating conditions or components change. However, due to the mixer and the turbopumps driven by the heated fuel from regenerative cooling, the high degree of coupling poses additional challenges to engine control, increasing the demand for advanced control methods.[6]

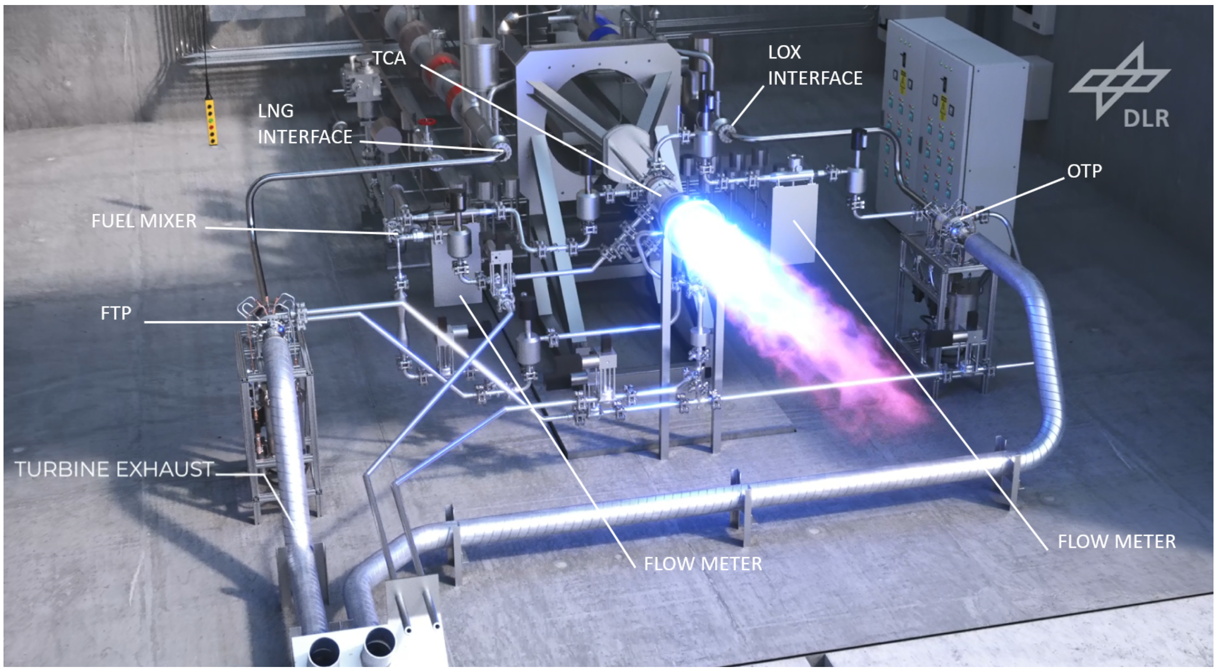


Figure 4.1: LUMEN at Test Bench P8.3 in Lampoldshausen, picture by courtesy of DLR

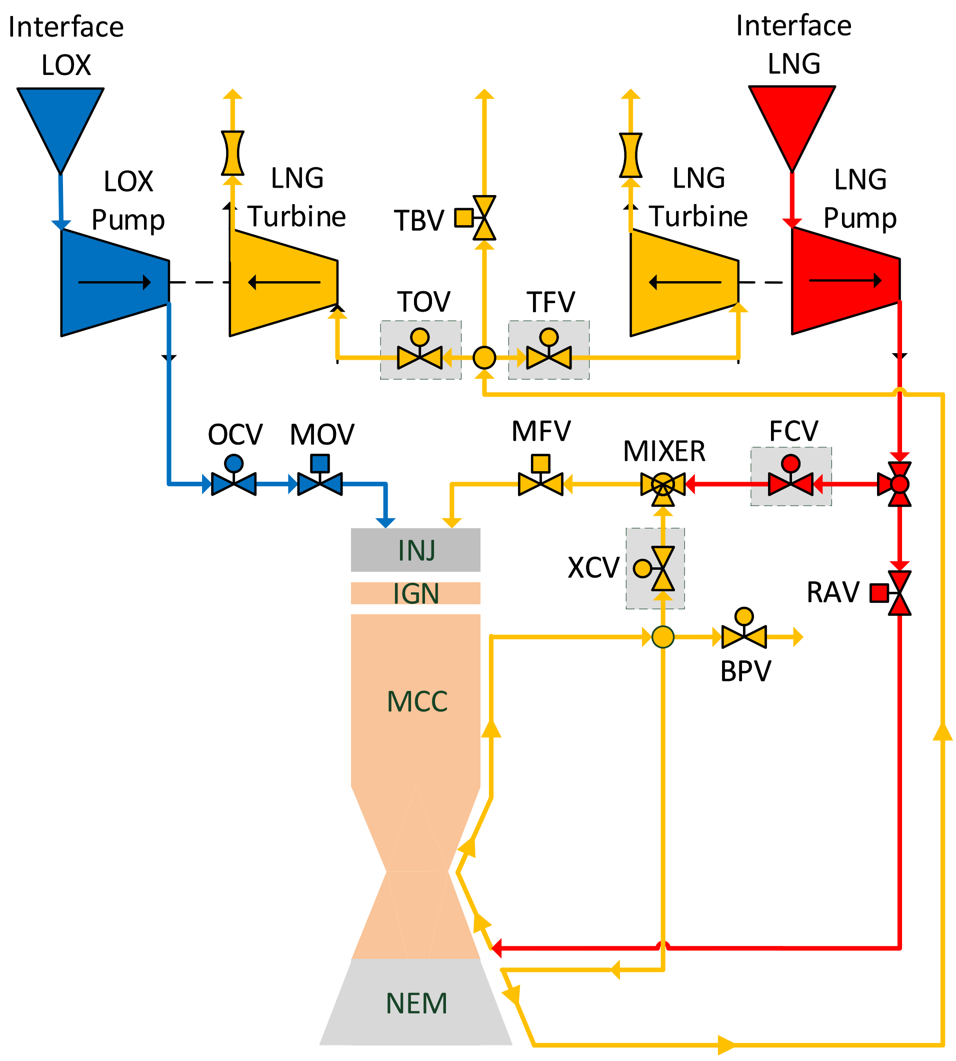


Figure 4.2: LUMEN Engine Architecture from Dresia et al.[6]

4.2 Operating Envelope and Constraints

The operating envelope of the LUMEN engine was chosen to be relatively large, with combustion chamber pressures from 35 to 80 bar and mixture ratios from 3.0 to 3.8 for LOX/LNG, resulting in a large throttling range of about 58 % to 133 % as required by the LUMEN design specification. The nominal operation point is set to a combustion chamber pressure of 60 bar and a mixture ratio of 3.4 (Table 4.1). This allows the investigation of technologies required for reusable engines and the integration and operation of a broader range of components.[18]
Constraints arise from operational requirements as well as maximum mechanical and thermal stress of individual components, including lifetime requirements.

Table 4.1: LUMEN Operating Envelope from Technical Specification Document [8]

Parameter	Unit	Nom.	Min.	Max.
Combustion Chamber Pressure	bar	60	35	80
Combustion Chamber Mixture Ratio	-	3.4	3.0	3.8
Fuel Turbopump Pump Discharge Pressure	bar	82.5	82.0	112.1
Fuel Turbopump Pump Mass Flow	kg/s	2.9	1.6	4.1
Fuel Turbopump Turbine Inlet Pressure	bar	35.4	30.2	70.2

In particular, the turbopump imposes limitations on maximum rotational speed, maximum turbine inlet temperature, and minimum pump inlet pressure (Table 5.3). If these limits are exceeded, there is a risk of turbine blade cracking, bearing or sealing damage, or the occurrence of cavitation, which damage the turbopump and, in the worst case, destroy the entire rocket engine.
Due to the properties of LNG and its usage as fuel and coolant, there are additional constraints for pressure and temperature at the injection and the cooling channels. The test bench P8.3 states additional constraints for LOX and LNG listed in Table 4.2. The limited tank capacity limits the total test time to 500 s, including chill down periods.[8]

Table 4.2: Test Bench P8.3 Interface Constraints from Technical Specification Document [8]

Medium	Mass Flow [kg/s]	Interface Temperature [K]	Interface Pressure [bar]
LOX	0.3-18	91±2	1.5-12.8
LNG	0-7	114±2	1.5-14.7

Table 4.3: LUMEN OTP and FTP Specification from Traudt et al.[9]

Parameter	OTP	FTP	Unit
Pump Pressure Rise	72	115	bar
Pump Power	80	214	kW
Rotational Speed	24000	50000	rpm

Condition monitoring and safety regulations are essential for successful testing of safety critical systems such as rocket engines. Besides red line logic triggering a controlled shutdown of the engine if one condition is not met, safety margins are an important strategy to operate technology demonstrators like LUMEN. For instance, one can increase the test bench interface pressure for cavitation mitigation but nevertheless the requirement for minimal pump inlet pressure should be monitored.

4.3 LUMEN Fuel Turbopump Test

In preparation for the LUMEN system tests in 2023, all subsystems are tested in individual test campaigns beforehand. Besides the design qualification and verification of workmanship, it is also necessary to specify the operating envelope and to investigate operational limitations.
During the LUMEN FTP test campaign, a simplified test setup is used to perform a characterization of the complete operational envelope. In contrast to the LUMEN operation conditions, the turbine working gas is nitrogen at almost ambient temperature instead of heated methane. Gaseous Nitrogen (GN2) drives the turbine, whereas LNG is used as pump fluid. Two valves are used to operate the FTP. The TFV controls the turbine mass flow and the resulting turbopump power upstream of the turbine. The FCV at the pump discharge side is used for pump mass flow and discharge pressure control. An additional orifice is used to imitate the expected pressure drop of the remaining fluid system including the mixer, injection and regenerative cooling channels. In order to test also a large portion of the LUMEN fluid system, the LUMEN test infrastructure including pipes, orifices and valves are being used already.
The test campaigns begin with cold gas flow tests before a set of different operating points will be approached with open loop sequences. These test results are beneficial for model validation and updates to select the appropriate controller or to modify the controller design accordingly. As neural network based controllers have to be trained, this may take a longer preparation time.

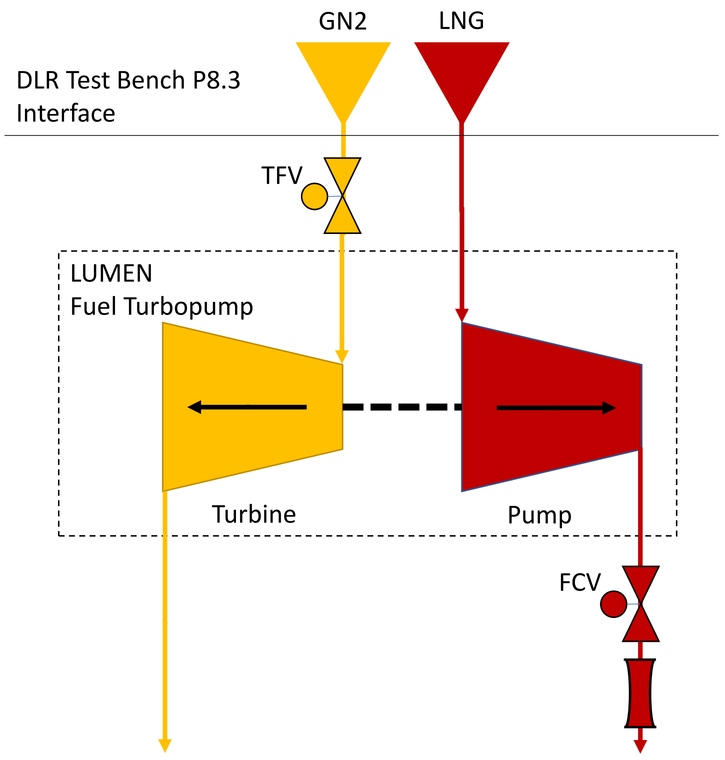


Figure 4.3: LUMEN FTP Test Setup

A preliminary risk analysis identified the main risks as: commanding failure or loss, sensor failure, controller malfunction and limitation violation. Except for the commanding interface, all failure causes are related to the controller. Due to the unavailability of test data, the controller design must rely only on numerical models. For this reason, apart from programming errors and faulty controller configuration, model uncertainties and external disturbances constitute the main risks for a successful controller design. As a consequence, high robustness requirements are necessary.
As this test entails no firing, the safety-critical requirements are significantly lower, so that modern control methods can be applied for the first time. By demonstrating the basic feasibility for this control task, the test marks an important milestone on the road to the future control of an entire engine.

Table 4.4: LUMEN FTP Test Conditions

Parameter	Value	Unit
LNG Interface Pressure	10	bar
LNG Interface Temperature	120	K
GN2 Interface Pressure	60	bar
GN2 Interface Temperature	273	K
Valve Delay	0.07	s

Chapter 5

Controller Design

This chapter deals with the design, configuration and training of the neural network based controller for the LUMEN FTP. As already introduced, the RL-based controller design needs to apply a set of tools to improve its robustness to cope with real-world challenges specific to the turbopump operation. After an overview of the EcosimPro/European Space Propulsion System Simulation (ESPSS) model of the LUMEN FTP and a model analysis concerning the turbopump performance and the model's sensitivity to parameter variations, the control objectives are defined. Together with the RL framework, the RL algorithm, and training configuration, the prerequisites for training the controller are discussed. The effects on robustness and control performance are evaluated for the RL environment including action space, observation space, reward function as well as robustness measures and the training procedure. In the following sections, the training with curriculum learning and domain randomization is further elaborated. Their effectiveness and performance are discussed on the basis of an in-depth evaluation of almost one hundred trained controllers listed in Appendix C Section C.2.

5.1 LUMEN Fuel Turbopump Model

EcosimPro is a well-validated system modeling and simulation software developed by the European Space Agency (ESA) similar to Matlab/Simulink. With its extensive ESPSS library of space propulsion components, it becames an indispensable tool for rocket engine system analysis [59]. An already implemented interface to Python can be used to export parameterized decks for platform-independent applications, customization and efficient model interactions enabling the usage for RL methods.

For the controller design, an already existing EcosimPro model of the LUMEN FTP was provided by Kai Dresia and Robson Henrique Dos Santos Hahn. The boundary conditions and constraints are investigated and a sensitivity analysis is performed to gain more detailed insights into the modeled turbopump system behavior. Because real-world test data are not available before the first FTP test campaign, this model analysis also serves as validation to reduce the model mismatch for the specified turbopump design. A plausibility check ensures model integrity. In addition, the EcosimPro model is also validated by checking all variables for the nominal LUMEN operating point against the design predictions and assumptions made by the turbopump expert group around Tobias Traudt and Robson Henrique Dos Santos Hahn. In Table 5.1, the nominal operating point is briefly characterized.

Table 5.1: LUMEN FTP Nominal Operating Point Characterization

Category	Parameter	EcosimPro Parameter	Value	Unit
Pump	Mass Flow	Pump_Fuel.m	2.9	kg/s
	Inlet Temperature	Pump_Fuel.f1.T	120.1	K
	Outlet Temperature	Pump_Fuel.f2.T	126.3	K
	Inlet Pressure	Pump_Fuel.f1.P	9.5	bar
	Outlet Pressure	Pump_Fuel.f2.P	82.5	bar
	Speed	Pump_Fuel.sh_in.n	33617.0	rpm
	NPSH	Pump_Fuel.NPSH	189.4	m
	Efficiency	Pump_Fuel.eta	56.6	%
	Torque	Pump_Fuel.sh_in.T	26.2	Nm
Turbine	Mass Flow	Turbine_Fuel.m	1.1	kg/s
	Inlet Temperature	Turbine_Fuel.f1.T	269.8	K
	Outlet Temperature	Turbine_Fuel.f2.T	178.4	K
	Inlet Pressure	Turbine_Fuel.f1.P	48.2	bar
	Outlet Pressure	Turbine_Fuel.f2.P	2.7	bar
	Speed	Turbine_Fuel.sh_in.n	33617.0	rpm
	Power	Turbine_Fuel.Power	92.2	kW
	Efficiency	Turbine_Fuel.eta	55.3	%
	Pressure Ratio	Turbine_Fuel.PI_tt	16.2	-
Turbopump	Mass	TP_mass	3.4	kg
	Moment of Inertia	TP_inertia	0.00235	kg·m^2

Throughout the test preparation phase, new experiences and measurement data from Oxidizer Turbopump (OTP) and Thrust Chamber Assembly (TCA) test campaign are contributing to model improvements, especially for the valve modeling and turbopump efficiency. Future research efforts shall focus on the application of ML for system identification to optimize the model with measured data [12]. Other RL concepts using online system identification or DA techniques require target domain data and may be considered after the first test campaigns.
In the EcosimPro model at hand, the FTP is embedded in the test setup described in Chapter 4. The entire setup includes identical pipelines, valves and orifices of the later LUMEN setup to test their functionality in advance. Valuable experience can be gained for the fluid system from their interaction on system level. The turbine and pump components are taken from the turbomachinery library and connected via an additional efficiency module to account for performance changes. The efficiency module scales the total efficiency of turbine and pump.

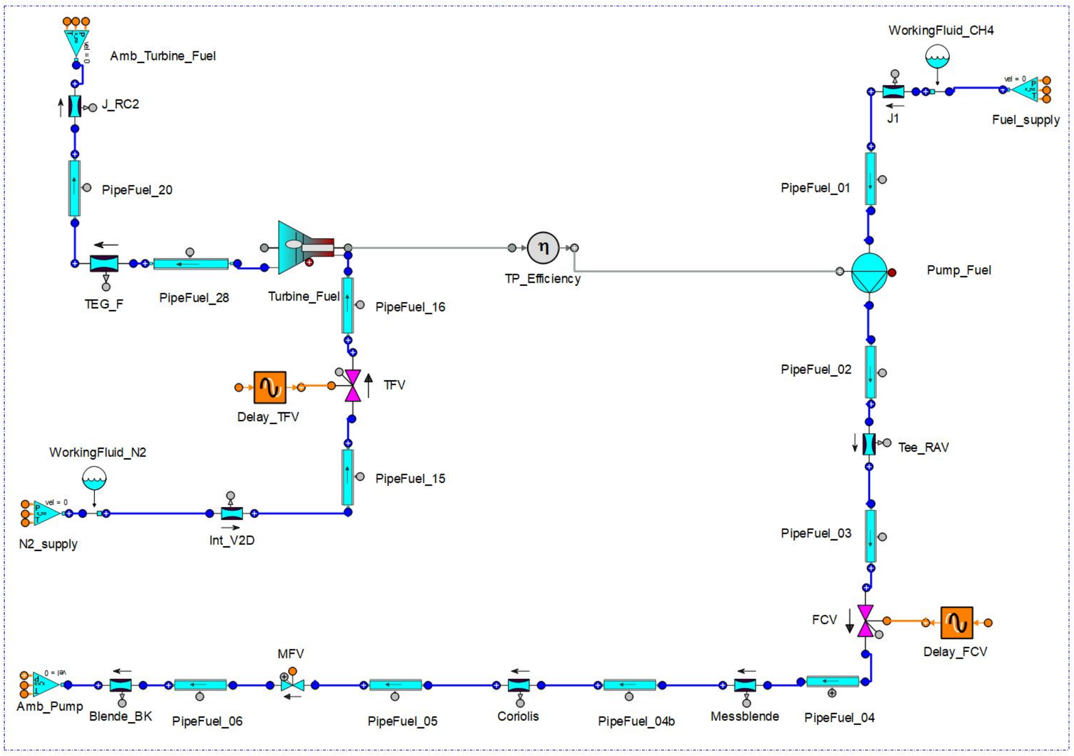


Figure 5.1: LUMEN FTP EcosimPro/ESPSS Model by courtesy of DLR

Pipe components are used to model the pipe system providing gaseous nitrogen and liquid methane from the test bench interface and discharging turbine exhaust gas and pump fluid. Custom valve components represent the respective valves from the test setup with individual configurations. The valve delay is customized by additional dead time components. For modeling of turbine and pump outlet, additional pipe components and orifices are used, which are relevant for the pressure losses and the associated efficiency.
The turbopump is defined with custom parameters, dimensionless characteristic curves and performance maps, which can not be transferred directly to the standard pump performance maps introduced in Chapter 2. Steady-state and transient cycle processes, such as start and shutdown, can be simulated. The EcosimPro/ESPSS user manual [60] is recommended for further information regarding the modeling of turbomachinery parts, pipes and valves.
An important part of modeling considers the different valve characteristics. Two electric valves are the actuators for the FTP control system. The Turbine Fuel Valve (TFV) controls the turbopump power by limiting the turbine mass flow and pressure ratio. The Fuel Control Valve (FCV) on the pump discharge side controls the pump mass flow and thus the turbopump load. As shown in Figure 5.2, both valves are necessary to control pump pressure rise and mass flow rate. In addition, the different valve flow characteristics become obvious as the TFV has a linear and FCV an equal-percentage flow characteristic. The flow characteristic is the change of the flow coefficient Cv as a function of the valve position.

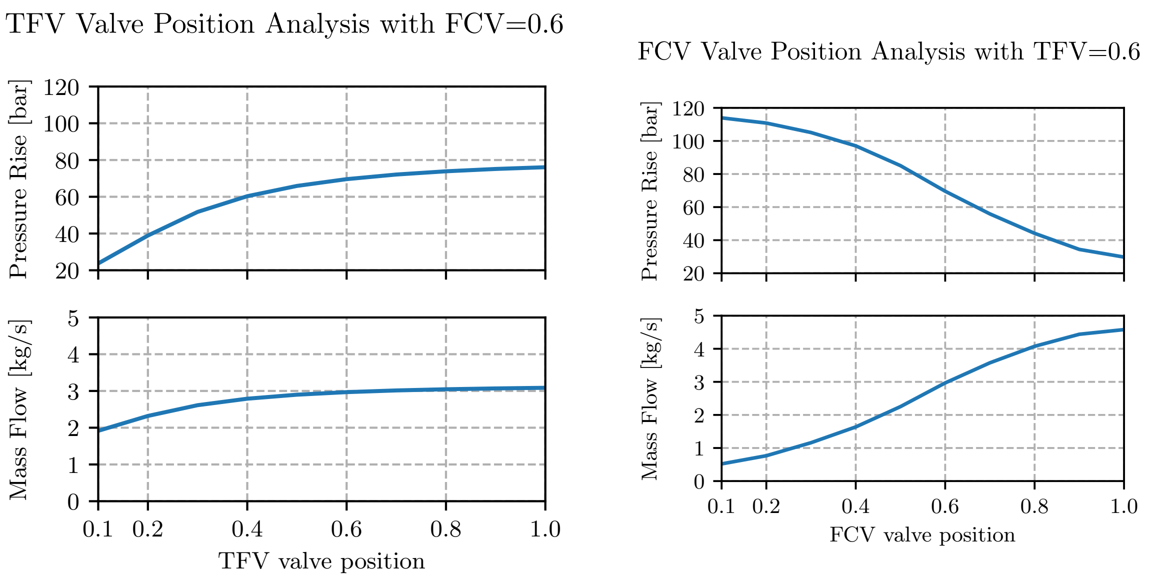


Figure 5.2: Valve Position Analysis

Moreover, the time-dependent valve behavior is difficult to model because the valve dead time depends on the commanded valve position change due to the inherent inertia and the valve servomotor control. The servomotor behavior has a certain acceleration, linear motion and deceleration phase. Both valves are modeled by a first-order transfer function where the delay parameter corresponds to the initial dead time and the max speed parameter determines the slope of the linear phase of the valve position change, as shown in Figure 5.3. Finally, the modeled valve characteristics are validated with manufacturer specifications and the first results from cold gas flow tests.

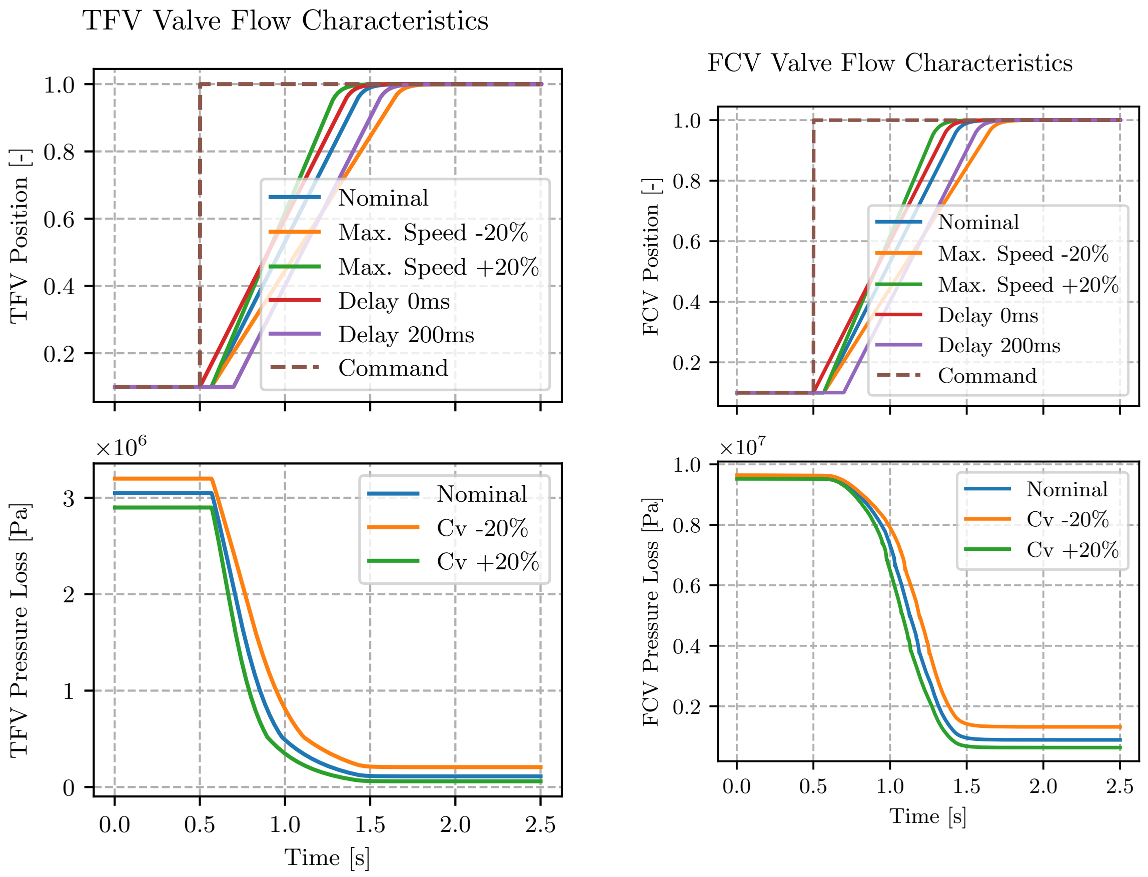


Figure 5.3: Valve Characteristics for a Step Command at 0.5 s

After an initial analysis of the transient response behavior for varying valve commands from different initial conditions, the analysis is extended to all settable model parameters. Within a model sensitivity analysis, the influence of each parameter on the system behavior in terms of steady-state error and settling time was examined. This analysis could be extended similar to the approach proposed by Pérez-Roca et al. [61].
In Table 5.2, the model parameters are summarized and it is indicated which pressure loss is changed with geometric modifications. Besides turbopump inertia and efficiency, test bench interface conditions and valve characteristics are assessed. To consider additional system pressure losses before and after flow control valves, the diameter of pipes is also varied. It becomes apparent that an additional system loss upstream of TFV corresponds to reduced GN2 supply pressure and has not to be analysed separately.

Table 5.2: Parameters for Model Sensitivity Analysis

Category	Parameter Description	EcosimPro Parameter
FTP Properties	FTP Efficiency	TP_Efficiency.eta
	FTP Inertia	Pump_Fuel.I
	Turbine Outlet Pressure Loss (Turbine Exhaust Gas Nozzle Area)	TEG_F.Ao
FCV Characteristics	FCV Delay	Delay_FCV.tdelay[1]
	FCV Speed	FCV.speed_max
	FCV Flow Coefficient	FCV.Cv
TFV Characteristics	TFV Delay	Delay_TFV.tdelay[1]
	TFV Speed	TFV.speed_max
	TFV Flow Coefficient	TFV.Cv
P8.3 Test Bench Interface	GN2 Pressure	p_GN2
	GN2 Temperature	t_GN2
	LNG Pressure	p_LNG
	LNG Temperature	t_LNG
System Pressure Losses	FCV Upstream Pressure Loss (Pipe 03 Diameter)	PipeFuel_03.D
	FCV Downstream Pressure Loss (Pipe 04 Diameter)	PipeFuel_04.D
	TFV Downstream Pressure Loss (Pipe 16 Diameter)	PipeFuel_16.D

In Figure 5.4, the results of the model sensitivity analysis are shown, which are crucial for the later controller design and robustness analysis. Test bench interfaces, valve characteristics and turbopump efficiency are the most important parameters determining the system behavior. Turbine parameters are directly influencing the turbopump power and cause significant deviations. Fluid temperature variations are affecting the fluid density, in particular for the pump medium LNG. Valve time characteristics are influencing the transient response behavior as expected. Depending on the relevance for the turbine or pump side, it is worthwhile to compare the effects on both reference values for pump discharge pressure and mass flow. The variation of the turbopump inertia and the system pressure losses have no major impact. The system pressure losses are several orders of magnitude smaller than the considered variations in supply pressures.
Since the model sensitivity analysis was carried out for the first operating point change of the reference trajectory defined later, a much smaller control variable change was considered for the FCV than for the TFV and thus the influences of the FCV valve characteristics are also significant smaller for the transient response behavior. As a result, the different perturbations due to different operating conditions can also be demonstrated.

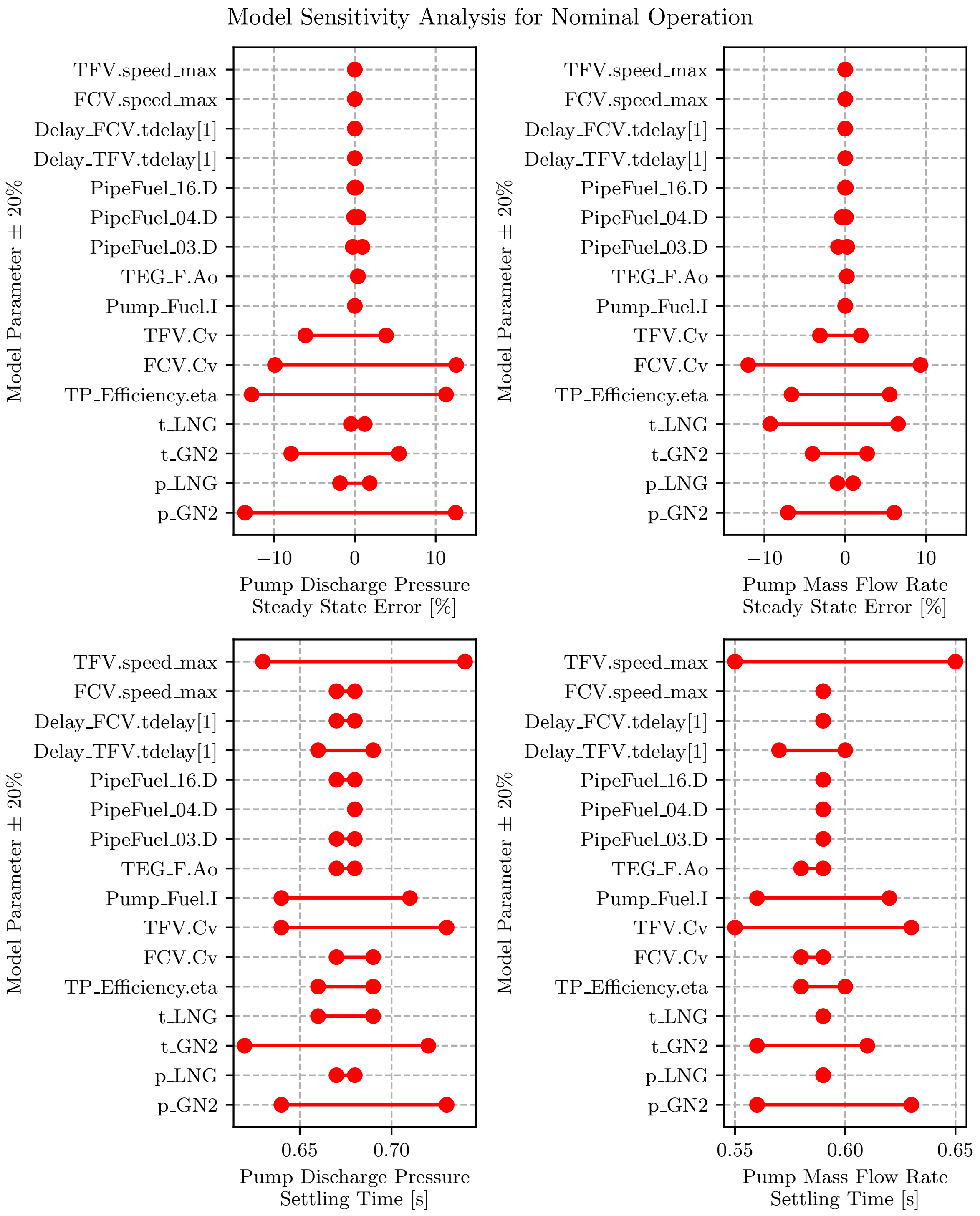


Figure 5.4: Model Sensitivity Analysis

Moreover, a gridding analysis was performed for TFV/FCV position combinations to estimate their effect on the turbopump operation as well as to conduct an initial plausibility check. Constraints and physical limitations are investigated and set up as termination conditions of the simulator. As the test bench interface conditions ensure that the maximal turbine inlet temperature and the required NPSH are always maintained for the specified operating envelope in section 5.2, these were not configured. The termination constraints are listed in Table 5.3.

Table 5.3: Termination Constraints for LUMEN FTP Simulator

Constraint	Value	Unit
Maximal Turbopump Speed	57000	rpm
Maximal Pump Discharge Pressure	200	bar
Minimal Turbine Inlet Pressure	1	bar

To analyse the turbopump operation in more detail, a pump characteristic map was created from a valve position gridding. Given the pump map, the required nominal valve positions can be determined for a certain operating point. To keep the reference to the technical specification of LUMEN, the pressure rise is plotted over the mass flow rate instead of pump head and volumetric flow. This is acceptable as an incompressible fluid can be assumed.
Finally, it should be mentioned that the contact with the turbopump working group and the test bench personnel at DLR was always essential, particularly for model updates and validation.
At this point, it should be added that in EcosimPro the pressure is converted to head with equation 5.1. The Total Dynamic Head (TDH) and Net Positive Suction Head (NPSH) are calculated according to Equations 2.6 and 2.11.

$$h = \frac{p}{9.806\rho_{\text{in}}} \tag{5.1}$$

5.2 Control Objectives

The design of the reference trajectory includes requirements for the study of control performance and operational requirements of the turbopump. From the LUMEN technical specification, the LUMEN operating points are drawn in the performance map of the LUMEN FTP, as shown in Figure 5.5, marking important points of interest. In particular, as these points are tested already prior to the FTP controller test campaign providing additional input to calibrate the final controller in preparation.
Figure 5.6 shows the reference trajectory created by connecting selected operation points. Step and ramp functions with rising and falling slopes combined with intermediate steady state plateaus enable the analysis of transient response performance for different aspects of the system dynamics. The set point for TFV=0.2 and FCV=0.6 at the beginning and end of the closed loop control ensures a safe start and shutdown with pretested open loop sequences without endangering operational constraints.

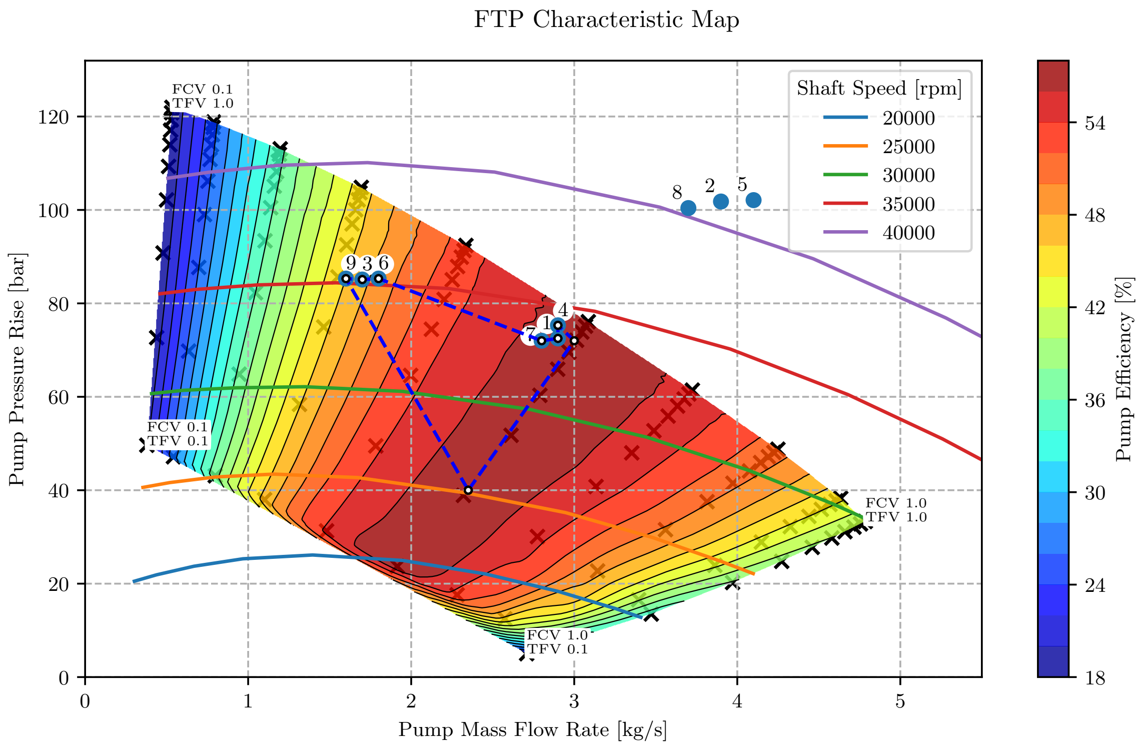


Figure 5.5: LUMEN FTP Performance Map with Reference Trajectory

To enable multiple tests per day, the duration of the trajectory is limited to 90 seconds. After 5 seconds of stationary operation in open loop control, the closed loop control takes over. After additional 5 seconds of stationary operation for compensation of deviations, the discharge pressure and mass flow are raised linearly. The pump performance chart shows that the speed is continuously increased with constant efficiency during this section. The reached operating point is held again to measure overshoots before the adjacent operating points four, one and seven are approached. This is followed by a step transition to the next group of operating points with operating points six, three and nine. In both groups of operating points, the first is held stationary for 10 seconds, and the second and third for 7 seconds. Considering the pump performance map, on the first ramp mainly the TFV valve has to be controlled to increase the turbopump power, and during the step, mainly the FCV valve is used to reduce the pump load. Over a ramp, the discharge pressure is reduced and the mass flow is raised so that both valves must necessarily be controlled at the same time and the starting point is reached. After a stationary operation of another 5 seconds, the turbopump is safely shut down. The flipped version of the reference trajectory allows the investigation of acceleration and deceleration of the turbopump in different operating areas.

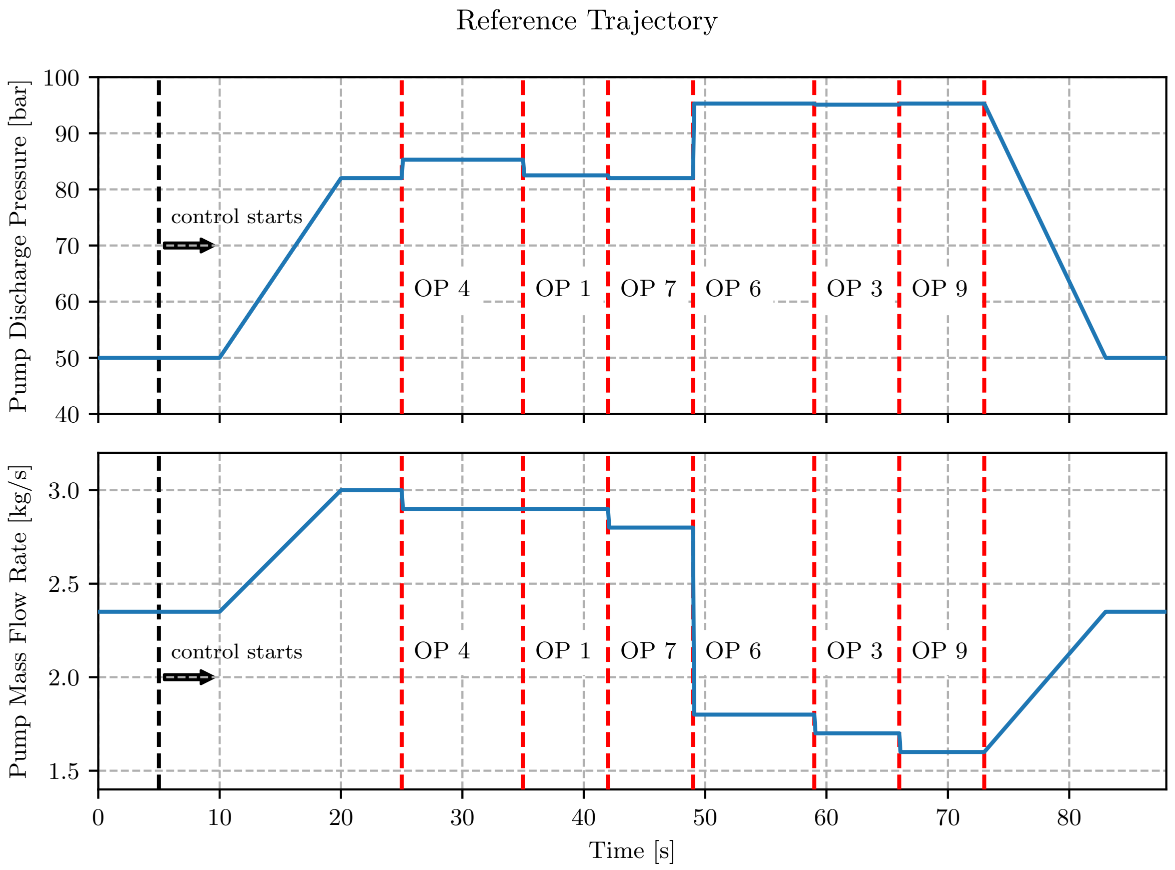


Figure 5.6: Reference Trajectory

Table 5.4: Control Requirements

Requirements	Criteria
Control Performance	Overshoot $M_p = \pm 10\%$
	Steady-State Error $e_\infty = \pm 5\%$
	Rise Time $T_r = 1\,\mathrm{sec}$
	Settling Time $T_s = 5\,\mathrm{sec}$
Stability	Demonstrate that oscillations are suppressed and damped
Robustness	Demonstrate that model uncertainties are compensated
Disturbance Rejection	Demonstrate that transients and oscillations are damped
Performance Deterioration	Demonstrate that time-dependent performance losses are compensated

The control requirements cover the minimization of the settling time below five seconds and the avoidance of overshoots. Furthermore, the steady-state error should be less than five percent. Since significantly higher accuracies are to be achieved with a closed loop control, the requirement $\pm 10\%$ from the actual technical specification, which considers an open loop control, was increased [8]. In addition, a stable operation must be guaranteed whereby oscillations are suppressed and damped. The control requirements are summarized in Table 5.4

5.3 Reinforcement Learning Framework

The RL problem must be defined within a suitable framework to train the neural network based controller. For Python, typically used in data science applications, Tensorflow, Gym and Ray are the main tools to set up the RL environment, to run the RL training and to analyse the achieved performance. The open source toolkit Gym [62] is widely accepted to set up RL environments for continuous and discrete spaces and to integrate RL algorithms from various libraries to learn the RL agent's policy. For future research acivities, Gym facilitates benchmarking of different RL methods with several challenging RL environments, for example the Real World Reinforcement Learning (RWRL) Challenge Framework from Google Research [63]. The Ray [64] framework is used for the RL algorithms and training as it supports distributed computing. Ray Rllib [65] provides the SAC algorithm and training utilities, whereas Ray Tune [66] comes with the tools for hyperparameter tuning.
A basic RL environment template could be reused from the research project on the non-linear control for LUMEN [6]. As the turbopump control task is completely different this time, only the main structure could be kept and had to be extended by numerous functions. The following subchapters address each part of the RL environment, including the robustness improving features crucial for the transfer to real-world application.
The overall goal has been achieved by developing a framework with a high degree of modularity and automatization to enable the reuse for future RL problems related to the field of control applications. This toolbox offers outstanding versatility thanks to the modules and configuration files that enable the extension of existing RL environments without much effort. In particular, the LUMEN project will benefit from the program code developed in this work.

5.3.1 Action Space

The action space comprises the definition of all control actuators. For the LUMEN FTP test campaign, these are two electric flow control valves to control the turbine inlet flow with the TFV and the pump discharge flow with the FCV. While the valve characteristics are modeled within the EcosimPro model, the RL environment defines the possible valve positions. For both valves, a continuous action space is defined with a limitation to positions from 0.1 to 1.0.
The absolute valve positions are used in contrast to the relative valve position change Hörger applied [16]. The proposed RL framework achieved limited performance with DR as the agent was probably unable to derive the relationships between absolute valve positions, given in the observation space, and the controlled variables and their deviations due to varied model parameters [16].

As already discussed, the first cold flow test results of the electric flow control valves showed a control input dependent valve dead time if the position change is larger than 5 %. One possible solution to tackle this issue can be to implement a clipping of the commanded valve position change. Another option is to increase the robustness to valve delays, which is intended here.
Standard RL algorithms optimize the parameterized neural networks to find the optimal policy and require normalized actions for better gradient-based optimization properties. A scale function is used to convert actions from the [-1, 1] range used by the RL algorithm to the RL environment [0, 1] range to compute the next state with the simulation model and the reward function accordingly. The action for time step t is defined in Equation 5.2.

$$a(t) = \begin{bmatrix} \mathrm{v}_{\mathrm{TFV,pos}}(t) \\ \mathrm{v}_{\mathrm{FCV,pos}}(t) \end{bmatrix} \tag{5.2}$$

5.3.2 Observation Space

The continuous, unbounded observation space comprises reference values for both pump discharge pressure and mass flow rate as well as valve positions and commands. Besides the reference values for the current time step, the reference for the next time step is always given. Additional future reference values can be customized and are generated for each time step from the interpolated reference trajectory, similar to the prediction horizon of MPC control. In contrast to Hörger [16], valve commands are included in the observation space to allow the agent to deduce predictions for the next states.
Another essential feature is the stacking of a number of past observations. With this measure, the agent gains the ability to access the state history to deduce information about the time-dependent system behavior. This approach was used by early projects [34][45], instead of recurrent neural network architecture, and is strongly related to online system identification from adaptive control. The ObservationWrapper class provided by Gym was adopted to the FTP RL environment structure to implement observation stacking. The observation for time step t is defined in Equation 5.3. The configuration parameter num_stacked_obs sets the observation history length and num_future_ref specifies the prediction horizon for the reference.

$$o(t) = \sum_{\tau=t-\mathrm{num\_stacked\_obs}}^{t} \begin{bmatrix} \mathrm{p}(\tau), \dot{\mathrm{m}}(\tau), \\ \mathrm{p}_{\mathrm{ref}}(\tau), \dot{\mathrm{m}}_{\mathrm{ref}}(\tau), \\ \mathrm{p}_{\mathrm{ref}}(\tau+1), \dot{\mathrm{m}}_{\mathrm{ref}}(\tau+1), \\ \vdots \\ \mathrm{p}_{\mathrm{ref}}(\tau+\mathrm{num\_future\_ref}), \dot{\mathrm{m}}_{\mathrm{ref}}(\tau+\mathrm{num\_future\_ref}), \\ \mathrm{v}_{\mathrm{TFV,com}}(\tau), \mathrm{v}_{\mathrm{FCV,com}}(\tau), \\ \mathrm{v}_{\mathrm{TFV,pos}}(\tau), \mathrm{v}_{\mathrm{FCV,pos}}(\tau) \end{bmatrix} \tag{5.3}$$

5.3.3 Reward Function

The multi-objective reward function computes a scalar reward value, as required by the MDP. Four different contributions are considered.

For trajectory tracking, a reward component is based on the absolute percentage control error. The sigmoid function proposed by DeepMind [13] is applied as transformation to improve the gradient-based learning even when the error is relatively small such that the learning is hampered, similar to the vanishing gradient problem. For a better understanding, the implemented sigmoid function is shown in Figure 5.7. In addition, a penalty term is added if the control variable is outside of a tolerance band of two percent around the trajectory.

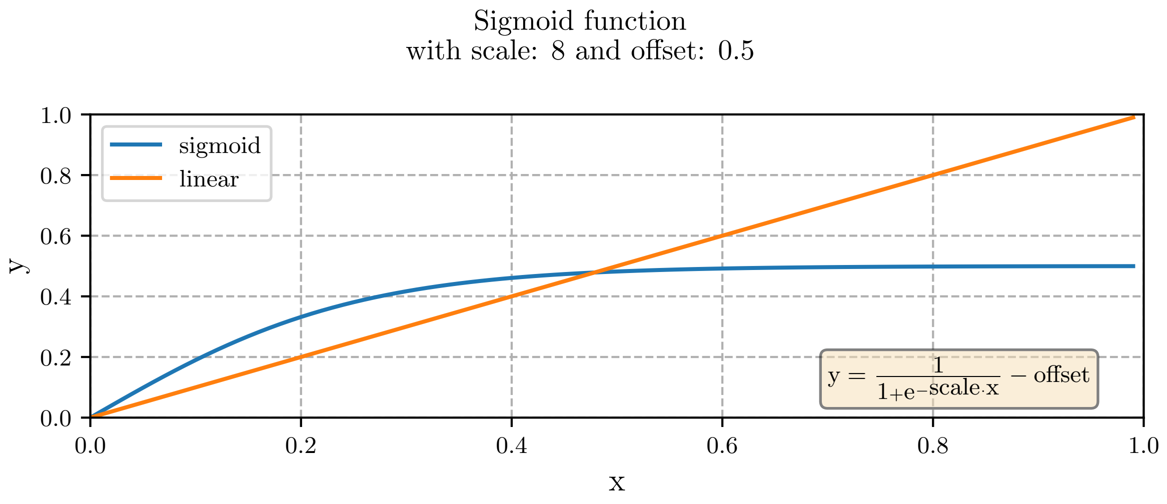


Figure 5.7: Sigmoid Function for the Tracking Error Reward Component

For trajectory tracking, the absolute control error could also be used to ensure similar absolute steady-state errors even for large set point changes. As for the reward function a normalization of both reference tracking errors is required, the absolute percentage error was used, accepting less performance for larger reference values. In particular, this solution is preferable to make the otherwise very abstract reward quantity more convenient, since the MAPE is an important performance measure in control engineering as well.
Moreover, stability and the required damping of oscillations is an important control objective. Therefore the difference between the current and previous actuator position is used as reward component. An additional reward term is computed from the difference between commanded and current actuator position to reduce overshoots. Intensive reward analysis showed that both terms related to the actuator position enforce the damping of oscillations.

$$r_{t+1} = -\left(\sum_{\mathrm{x} \sim [\mathrm{p}, \dot{\mathrm{m}}]} r_{t+1_{\text{tracking}}} + \sum_{\mathrm{v} \sim [\mathrm{TFV}, \mathrm{FCV}]} r_{t+1_{\text{actuator}}} \right) \Delta t \tag{5.4}$$

For later reward analysis, the reward function is parameterized by the RL environment configuration, including individual gains and the tolerance specified for the tracking error penalty. In Equation 5.5, the reward function is shown where all α parameters are part of the configuration.

$$\begin{aligned} r_{t+1_{\text{tracking}}} &= \alpha_{\text{gain, tracking}} \cdot (f_{\text{sigmoid}}(e) + r_{\text{penalty}}) \\ \text{with} \quad e &= \text{ape} = \|\mathrm{x}_{\text{ref},t} - \mathrm{x}_t\| \\ \text{and} \quad r_{\text{penalty}} &= \begin{cases} \alpha_{\text{penalty}}, & \text{if } e \geq \alpha_{\text{tolerance}} \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{5.5}$$

$$\begin{aligned} r_{t+1_{\text{actuator}}} &= \alpha_{\text{gain, u}} \cdot \|\mathrm{v}_{\text{com},t} - \mathrm{v}_{\text{pos},t}\| \\ &+ \alpha_{\text{gain, command}} \cdot \|\mathrm{v}_{\text{com},t} - \mathrm{v}_{\text{com},t-1}\| \end{aligned}$$

Dulac-Arnold et al.[17] proposed the analysis of the reward function and each of its contributions to gain further insights about the trade-offs made by the agent. For this purpose, the individual components are depicted in one diagram. During the controller design, the advantages of this analysis became evident as the tuning of gains can be deduced from the reward composition of test runs. Low reward contributions have good learning properties as these were reduced during training compared to the baseline solution. Reward components, which were also low for the baseline controller, are not meaningful for this RL problem or appear only in case of disturbances not present in the baseline case, for instance oscillations caused by additional system delays. One example is presented in Appendix B
With regard to the selection of the control frequency, the reward is scaled by the time step size Δt (Equation 5.4). As the closed loop control starts delayed, the reward is computed only for the control period of 83 seconds. Given the control period time, the reward function and the configuration parameters, the worst-case expected reward can be determined for validation purposes.
If the agent is prone to explore impossible physical states normally prevented by proper selection of DR boundaries, it can be useful to add an additional penalty for specific termination conditions. About the control of the complete LUMEN engine, this should be considered as the control engineer may be unable to identify meaningful DR limits to avoid undesired states because of the high complexity and coupling. Moreover, this can be wanted regarding safe RL methods to provide the agent information about critical, inaccessible states.[13]

5.3.4 Robust Tools

The robust tools describe a set of methods to change the RL environment with the goal of improving the robustness of the RL agent's policy. A robust version of the RL environment inherits all methods and attributes of the FTP specific RL environment to increase the modularity and extends the functionality accordingly. Besides DR, action and observation delay and noise are implemented. As discussed in section 3.5, there are already different solutions to deal with each aspect. Here the implementation is described before in the following chapter several trainings are carried out to evaluate the effectiveness of each measure to tune the hyperparameters for the training of the final controller afterwards.

In subsection 3.5.1, different methods of Domain Randomization (DR) were discussed. The DR is realized with simple get and set methods to modify a specific model parameter during the reset of the RL environment. The sampling is based on a truncated normal probability distribution with adjustable standard deviation necessary for curriculum learning. The shape of the probability distribution is defined by the standard deviation and the domain parameter boundaries as the mean corresponds to the predefined nominal value. As shown in Figure 5.8, small changes in the standard deviation cause large differences until a uniform probability distribution is approximated for a standard deviation of 2.0. From statistics it is known that if the domain parameter boundaries are ±20%, the main influence of the changed standard deviation occurs between 0 and 20 %.

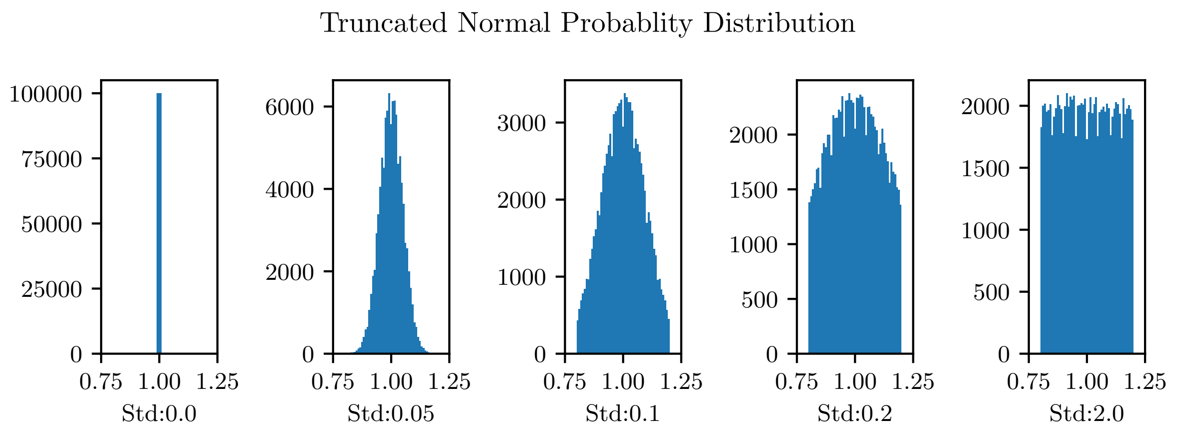


Figure 5.8: DR Sampling Distribution for different Standard Deviations

Noise can be applied to actions and observations to account for the jitter of actuators and sensor noise. Besides uncorrelated Gaussian noise, time-correlated noise is also implemented based on the Ornstein-Uhlenbeck process [40] from the DDPG algorithm by OpenAI [36].
Moreover, system delays have to be considered. On the one side, the valve delays are part of the domain parameters and can be arbitrarily varied even for a fraction of the control time step. On the other hand, actions and observations can be delayed by a number of integer time steps within the RL framework configuration.
In addition, a distinction is made if the delay remains constant or varies over an episode. Before each episode, the delay is randomly activated to guarantee that the baseline case can be sampled in the course of training[48].
Additional effects such as sensor offset or linearity errors are not considered. However, these should be investigated with regard to the more complex control of the LUMEN engine as individual effects may accumulate.

5.4 Training of Robust Control

During training, the RL agent's policy approximated by a neural network is optimized to maximize the expected cumulative reward for the given reference trajectory, the specified reward function describing control performance requirements and the LUMEN FTP specific RL environment. Because of its robustness to hyperparameters, sample efficiency and applicability for continuous action and observation spaces, the SAC algorithm is used. The Ray version of the SAC [4][7] is chosen as the extensive libraries allow user-customized, distributed training to be performed without much effort.
The actor and critic network structure with two layers, each with 256 Rectified Linear Unit (RELU) neurons, is not modified as this is out of scope for this master thesis project. In future research activities, the neural network architecture should be reviewed and studied not only with respect to size and extensions to RNNs, including LSTM concepts, but also for meaningful differences between actor and critic network design. For example, an omniscient critic can be advantageous, like Peng et al.[45] suggested. In particular, the usage of a history with past observations raises the dimensionality of the RL problem to such an extent that the neural network can be driven into saturation as the information capacity is limited. This may deteriorate the agent's learning ability.
Ray provides a set of hyperparameters to customize the RL training and algorithm configuration. Within an initial study of the baseline training (without DR), the trainer settings learning rate, discount factor and the SAC specific target entropy parameter are varied. With regard to improved convergence properties, the learning rate and discount factor are slightly decreased (learning rate of 1.5e-4 and gamma of 0.9) and the target entropy increased. The target entropy influences the choice of the temperature parameter that weights the entropy part of the target function. Normally the target entropy is automatically set to the dimensionality of the action space for the FTP RL environment -2, but it is increased to -3, thus enforcing more exploration. The standard training parameters are part of the Appendix C.
The training progress can be tracked with TensorBoard, which shows the evolution of different metrics over the number of training iterations. Actor and critic loss are important measures to check if the learning is still progressing or converged such that no significant improvements are expected. In addition, the maximum, minimum and mean training and evaluation episode reward can be inspected. These are required to control the progress of the curriculum learning addressed in the following chapter.
Since the evaluation reward provides information about the agent's performance, this parameter is used to select the best checkpoint and thus the trained policy of the neural network based controller. As the environment changes randomly in DR, a larger number of evaluation episodes must be run in order to obtain stochastically relevant information about the performance.
Finally, the training architecture shall be briefly discussed, focusing on parallelization and timing to follow the subsequent discussion regarding different settings. During training, the trainer operates locally the learner and replay buffer, including the optimization of actor and critic neural networks. A customized number of workers, processing units, is used to sample training and evaluation episodes. Depending on the timesteps per iteration and the checkpoint frequency, a certain number of episodes, also called rollouts, is performed until the policy is re-evaluated and stored as a checkpoint which allows testing and deployment as a controller later on.

The computer scientist is responsible to find the best setup considering computational resources, speed depending on the amount of parallelization and the ratio between training and evaluation iterations. In the future, Ray's full checkpointing capabilities should be used to be able to restart a training from any checkpoint so that in the event of computer crashes or updates, the training will not be lost.

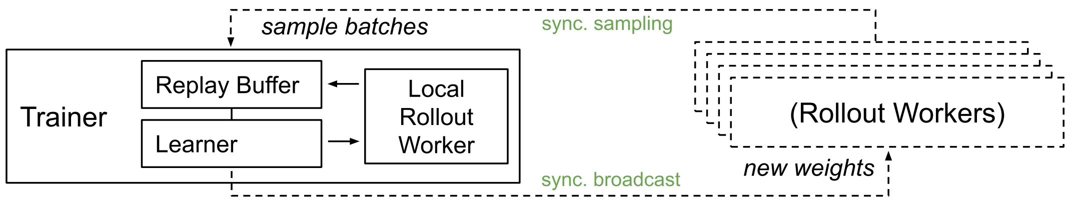


Figure 5.9: SAC Architecture from Ray Rllib [7]

In classical control engineering, the controller design consists mainly of model linearization using system identification methods, setup of complex state space models and an extensive tuning of the controller. Using RL methods, the control engineer is mainly concerned with the setup of the RL environment, including reward function shaping, the selection of a RL algorithm and hyperparameter tuning, and evaluating the performance.
Since in this work the training setup for the RL problem of the LUMEN FTP control task has been developed completely from scratch, a large part of this work is also dedicated to the study of the effectiveness of the robust tools presented in the following subchapters as part of the controller design. For this purpose, a series of trials with almost one hundred different configurations was planned and performed in order to study each aspect in more detail using a specially trained controller and to evaluate them in direct comparison with other controller designs. In Appendix C, the trainings are listed.

5.4.1 RL Environment Configuration

The already introduced RL environment can be configured with a specific configuration file, which is essential to reload and evaluate a trained policy later on. Before training of the baseline controller, a trial series (ID 010-015) was carried out to configure the reward function, control frequency and observation space accordingly. It could be shown that the reward components related to the actuator commands and positions are crucial for damping of overshoots and oscillations. The tracking error penalty with respect to a specific tolerance band reduces the steady-state error significantly, accepting less training stability. Adding future reference values improves tracking performance at the ramps but significantly lowers the transient response behavior by increased rise time. With regard to the derivatives of the trajectory, a number of three future reference values enable the agent to deduce information about the second derivative to follow the ramps without oscillations making the best compromise. The usage of stacked observations is not necessary to train a good baseline controller but a second run already incorporated DR and demonstrated the power of this tool.

The agent was able to derive system dynamic properties to counteract model parameter variations. For the baseline controller a number of ten stacked observations is used.
Finally, the control frequency, thus the time step, has been analysed and showed that a control frequency of even 1Hz is sufficient to control the FTP with the required accuracy. Considering the control objectives, including a fast transient response behavior, it was decided to design a controller for 10 Hz, which is supported by the test bench checkout system. The results of this investigation are summarized in Table 5.5.

Table 5.5: Investigation of RL Environment Configuration

Trials	Parameters	Benefits	Potential Drawbacks
ID 010	reward function actuator penalty	reduce oscillations	increase steady-state error
ID 011	reward function control penalty	reduce oscillations and overshoots	increase steady-state error
ID 012	reward function tracking penalty	reduce steady-state error	increase oscillations
ID 013	control frequency	reduce rise time	increase oscillations and overshoots
ID 014	number of future reference values	improve tracking and reduce oscillations	increase susceptibility to valve delays and oscillations prior to set point changes
ID 015	number of stacked observations	increase robustness to model mismatch and improve tracking	increase rise time

At this point, the results of the baseline controller are already presented to prove that the neural network based controller is able to follow the reference trajectory. In Figure 5.10, the agent reacts quickly to steps and moves precisely on the ramps using different movements of both valves TFV and FCV. In RL, this proof is often referred to as the sign of life. The resulting valve sequences can already be used in open loop control to operate the LUMEN FTP.

5.4.2 Curriculum Learning

The training scope is gradually expanded with curriculum learning in order to equip the neural network based controller with different capabilities by ensuring continuous learning. At the end of every training iteration, the mean evaluation episode reward is used as a measure of the agent's performance, and the next training level is activated if a specified threshold is exceeded. Robust tools like noise or delays can be activated but also parameters of the RL environment can be changed. The thresholds are sensitive hyperparameters by limiting the learning for a specific training level as soon as the next task is activated.

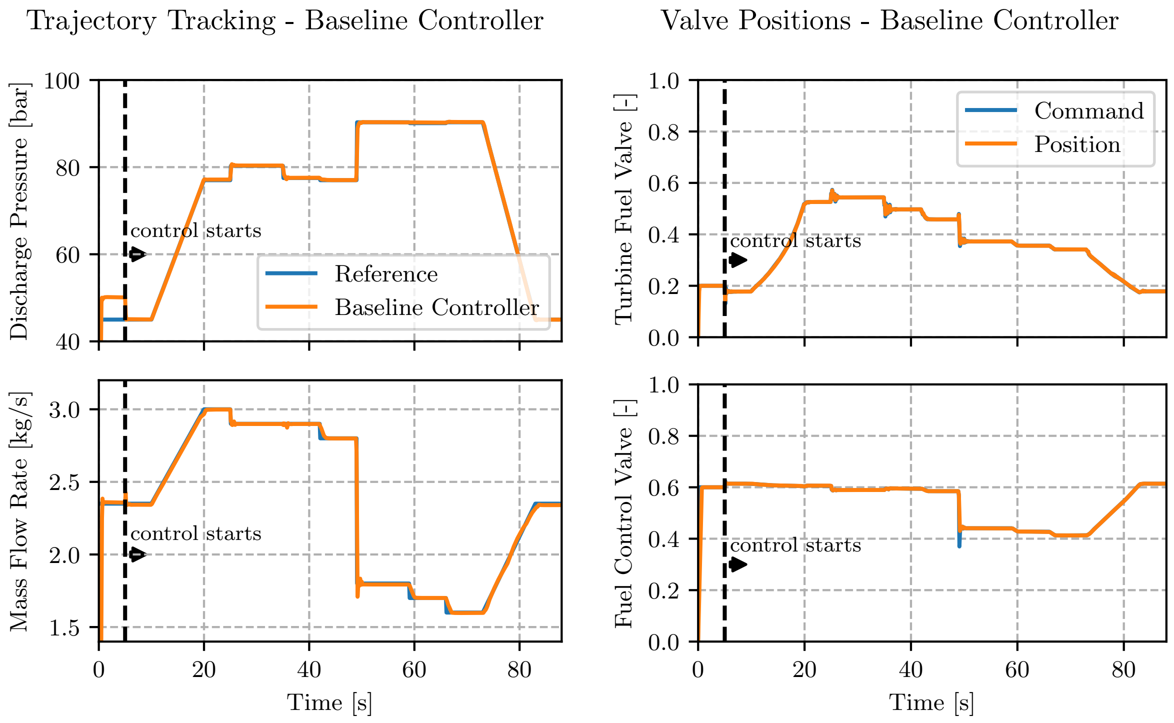


Figure 5.10: Baseline Controller for Nominal Case

In order to reduce this dependency, an additional condition for the duration of a training level can be specified. A counter of threshold crossings is used to specify the minimum time for convergence using a minimum constraint. Since the probability of reaching the threshold varies considerably and tends to fall with the appearance of new disturbances, a maximum constraint for the number of training iterations per training level can also be set to ensure that the training continues.

The mean of the evaluation episode reward is used for scheduling as this metric is a measure of the current policy's performance for the average of randomized environments. The maximum reward is a measure for the performance on environments close to the baseline case in contrast to the minimum, which serves as an indicator for the worst case performance comparable to the control objective of robust control.

By means of a fine-tuned training curriculum, the structure can be customized to the demands of a specific control task. It is possible to change parameters, thresholds and constraints during the training, which takes modularity, automatization and flexibility to the next level.

In Figure 5.11, an exemplary training procedure is shown. After the agent has learned to solve the nominal model well enough that a certain threshold is exceeded, the automatic DR is switched on. The procedure of the DR is explained in the following chapter. In the next level with increasing standard deviation more difficult environments are sampled, which leads to drops in reward until the threshold is reached and the standard deviation is adjusted again.

Then if the DR is completed, it can be specified that the DR is continued for 40 iterations with better rewards than required by the threshold for specializing the agent before noise is switched on. Since the baseline solution is also influenced by the noise, the reward will initially drop very sharply, so that it may be necessary to continue training after a maximum of 60 training iterations and to switch on the additional delay if the threshold value is not reached.

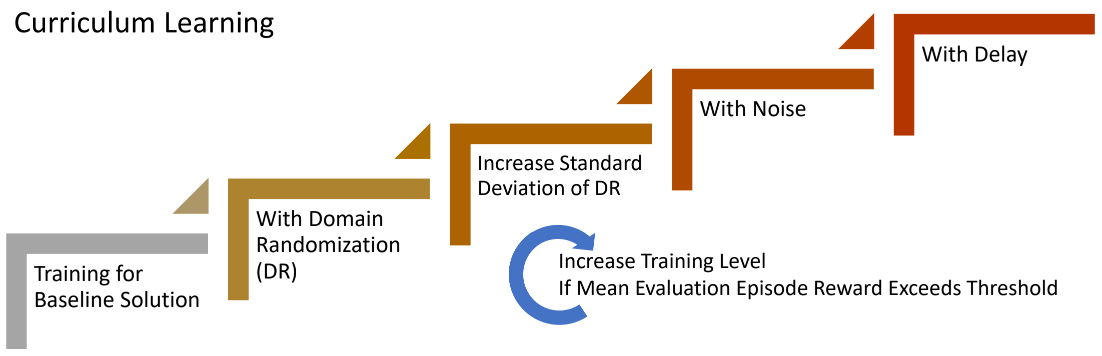

Curriculum Learning
Training for Baseline Solution
With Domain Randomization (DR)
Increase Standard Deviation of DR
With Noise
With Delay
Increase Training Level If Mean Evaluation Episode Reward Exceeds Threshold

Figure 5.11: Curriculum Learning

5.4.3 Domain Randomization

Similar to the Automatic DR proposed by OpenAI [50], the domain parameters are sampled from a reward-dependent changing probability distribution. Here a new concept was developed to start with a normal distribution which approaches a uniform distribution by a stepwise increasing standard deviation. As more likely more randomized environments are sampled, the training difficulty level increases. The Automatic DR with extendable DR boundaries requires more tuning effort for each domain parameter, which was not reasonable for the number of domain parameters, in particular with regard to the future control of LUMEN.
An extension towards a reward-dependent optimization of the probability distribution was developed so that parameter spaces leading to lower reward are sampled with higher probability. As this approach has a high risk of finding a conservative solution that achieves only average performance for all environments, it was not implemented and tested. Other concepts like the Active DR [49] or Adversarial DR [54][55] were also discussed and should be studied in follow-on research projects.
Comparing different strategies regarding the increase of standard deviation during the DR training phase showed that according to the probability distribution, the difficulty level changes the largest during the initial phase. For this reason, only a small standard deviation step size of 0.05 is used at the beginning and set to 0.2 after a threshold for the standard deviation of 0.2 is reached. This adaptive step size was compared to a constant step size in a trial (ID 044) and proved no remarkable difference, so the more efficient adaptive technique is chosen. Furthermore, a delayed model randomization of a secondary set of domain parameters was used to investigate their effect detached and to use an extension of the domain parameter set as an additional measure to control the training difficulty level. This measure can be beneficial for more complex RL problems but was not needed here finally.

Based on the LUMEN FTP model sensitivity analysis, it is already known which parameters can be expected to worsen the agent's reward the most if no countermeasures are considered during training. These parameters are suitable for the DR parameter set and their randomization boundaries are set according to the FTP design margins or, if unknown, with safety margins of $\pm 10 - 20\%$ typical used at this project stage. The domain parameter set with nominal, minimal and maximal values is part of Appendix C. Regarding systems engineering, the need for a margin philosophy has been recognized and will be developed in the next time.
To gain more insights about the relevance of individual model parameters used in randomization and the policy's robustness, a training series (ID 020-027) was performed each considering only one domain parameter in DR. Finally, all agents are compared to the baseline controller (ID 001) and the controller trained with the complete domain parameter set (ID 028) for the trajectory tracking performance of both reference quantities.
Here it should be mentioned that model updates regarding the turbine outlet pressure loss required the addition of the turbine exhaust gas outlet cross-section area as domain parameter studied in ID 052 and compared with the complete controller, thereafter also called "Complete Controller", in trial ID 054 for 5 Hz and ID 055 for 10 Hz.
The results are presented in Chapter 6.

5.4.4 Noise and System Delays

Uncorrelated Gaussian noise and correlated noise based on the Ornstein-Uhlenbeck process are studied for both observation (ID 030) and action (ID 031) noise. By reduction of hyperparameters, the complexity can be reduced. For this reason, sensor or actuator specific standard deviations normally used in classical modeling are replaced by standard deviations in percentage. An average standard deviation of 2 % for sensors and 1% for valves has been estimated from available test data. With the configuration file for the robust RL environment, noise can be activated for certain observation or action variables. The evaluation showed that in particular the noise in the commanded valve positions was damped and thus the sensitivity to noise could be reduced. The evaluation was mainly performed by visual inspection, since steady-state errors of less than 0.5 % caused considerably distorted metrics.
The study of system delays comprises the consideration of valve modeling, real-time capability and timing analysis for commanding and monitoring at the test bench. With regard to the RL problem formulation, system delays can be introduced for actions and observations as well as domain parameters for individual valves. The observation (ID 032) and action (ID 033) delay can be configured as an integer number of time steps depending on the selected control frequency. In the present implementation, the complete set of actions or observations is delayed without any differentiation referring to the specific actuator or sensor type. To enable different action delays, the valve specific delay and speed of TFV and FCV are included in the set of domain parameters for DR. Here, it has to be considered that both action delay and valve delay parameter can add up to unrealistic large dead times, which have to be avoided by properly tuned training schedules. Since system delays destabilize controlled systems due to oscillations induced at set point changes, the ISE control metric is used to measure overshoots and oscillations in particular. In Figure 5.12, the ISE is compared for the baseline, complete, constant and random observation delay and the final controller.

It is evident that the DR already improves robustness to observation delays but additional considered sensor delay improves even more. The random observation delay randomly selects per time step how large the delay is and performs slightly worse, but can be used as a preparatory step for the constant observation delay for curriculum learning of the final controller.

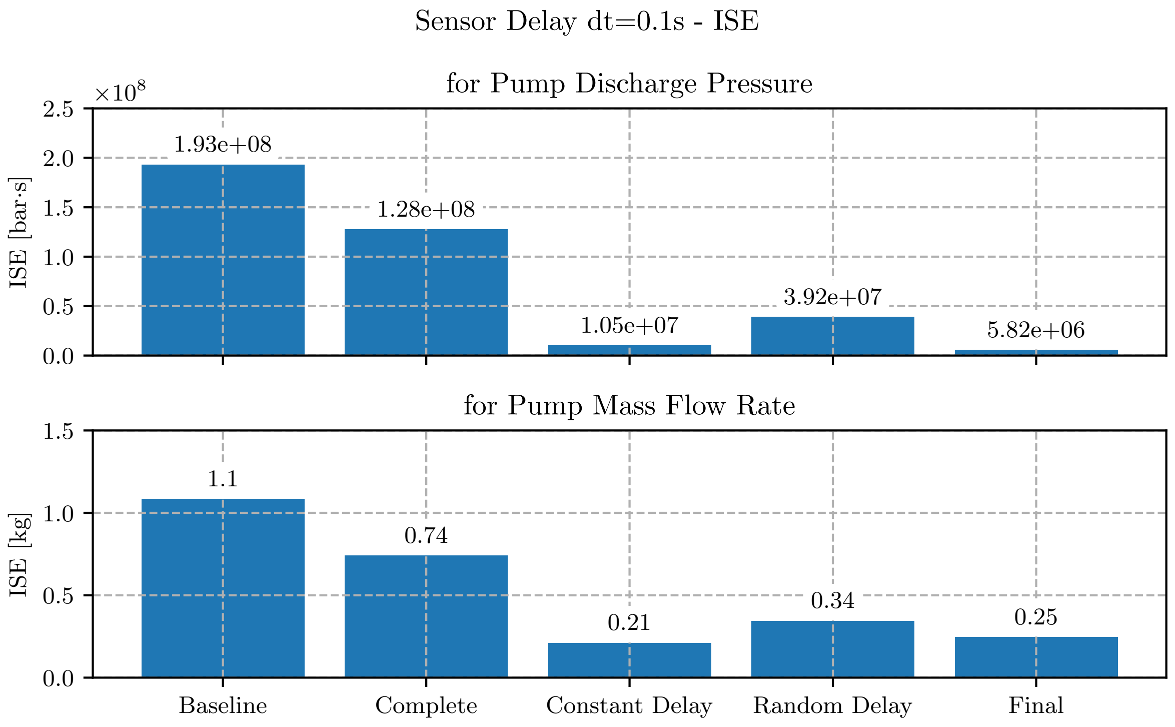


Figure 5.12: Observation Delay of One Time Step

Commanding and monitoring are tested parallel to rocket engine system tests and first results estimate a total time delay of 1ms for the computations of the neural network based controller and 25 ms for the commanding.
Time-dependent valve characteristics are modeled using first-order transfer functions and are prone to model uncertainties. Valve test data provide the foundation to improve the EcosimPro model by increasing the nominal valve delay from 10 ms to 70 ms. Even with unchanged DR boundaries, a significantly greater robustness to valve delay variation is obtained with the modified baseline solution trained at the beginning, once again emphasizing the relevance of a well-validated model. The valve test data required an extension of valve delays which were assessed in an additional trial series (ID 050 and 051). As the extended valve delay boundaries of 0 to 190 ms exceed the time step of 100 ms for a control frequency of 10 Hz, the control frequency of 5 Hz has been evaluated in parallel and showed large robustness improvements, which motivates the usage of a 5 Hz controller during the FTP test campaign. Nevertheless, both controllers are trained and will be available for testing.

5.4.5 Different Observability

Different observation spaces are studied to find the best trade-off between performance and complexity resulting in a longer training duration. Besides the number of future reference values (ID 040) and stacked observations (ID 041), the inclusion of additional sensor measurements (ID 042) and the exclusion of valve commands (ID 043) are investigated.
The stacked observations are crucial for the training success when using DR, as otherwise the agent is not able to learn the system dynamics. More past observations improve the robustness but reduce the transient response control performance as the settling time increases. This also applies to the number of future reference values essential to reduce oscillations. A possible explanation is the agent's ability to learn from the prediction horizon that oscillations are not useful as long as no set point change can be expected. The changed transient response behavior supports these considerations as the agent excites the system to oscillate shortly before a set point change. With regard to the later FTP operation, this may be restricted so that the number of future reference values has to be reduced accordingly. In addition, this is an important fact for the analysis of ITAE as the stationary behavior will be weighted. Moreover, a larger number of future reference values decreases the robustness to valve delay variations and increase the steady-state error in some operating areas.
Additional sensor measurements of test bench interface conditions such as pressures or temperatures are available during testing and can be used for the observation space. In contrast to the expectations, the additional state information worsens the success of the DR. It can be argued that these additional parameters are varied per episode and have no time-dependent information so that the agent may overfit to episode specific features, which degrades the generalization capabilities to unseen randomized environments.
The analysis of agents with and without valve commands as part of the observation space showed that the agent is not able to deduce the future valve positions from the future reference values of pump discharge pressure and mass flow. The usage of valve commands considerably improves the overall control performance. Hörger [16] did not incorporate valve commands and past observations, which could explain the missing success of the DR method.

5.4.6 Hyperparameter Tuning

Hyperparameter tuning is concerned with the customization of RL environments and RL algorithms to optimize the agent's policy while being computationally efficient, especially during training. Here the reward function gains, number of reference values and stacked observations are considered with respect to the training of the baseline as well as the final robust controller.
The extensive study of different RL setups showed that incorporating past observations is the key to successful DR leading to improved robustness to model uncertainties, disturbances and deterioration. In addition, an increased number of future reference values improves trajectory tracking performance while the settling behavior is slightly worse. In addition, oscillations are significantly reduced.
The configuration of the reward function results in different trade-offs made by the RL agent. The control engineer is mainly concerned to achieve all control objectives with maximal performance and robustness by appropriately tuning the reward function.

Nevertheless, it has to be assumed that the model will be updated during the test campaign so that parameters not considered during training may be changed. A modular RL framework and the knowledge of the significance of individual functions are important to react with appropriate measures to repeat the training. For example, after completion of trial series ID 04X, the turbopump modeling has been changed to account for new design analysis results. The controller was able to handle this model deviation without new training, even though the steady-state error for one operating point increased to almost two percent, which still satisfies the control performance requirements. The previously introduced explanation of inherent robustness is also applicable here and explains the robustness even to previously unknown model changes as the neural network based control can generalize.

Due to the late model update, the final controller was tuned again in a series of trainings (ID 050-055 and ID 100-105). Thereby, the focus is set on minimizing the steady-state error in preparation of the LUMEN FTP controller test campaign. In order to achieve this, the part of the reward function for the trajectory tracking is weighted more strongly, whereby the sensitivity to system delays increases slightly. This is acceptable as it does not cause performance losses for the already tested system delays of less than 150 ms. Furthermore, the training duration should be reduced to allow short-term training during the test campaign. A relevant factor is the prediction horizon, the number of future reference values, where a reduction from five to two values halves the time needed for DR. The potential disadvantages of increased overshoots and oscillations are not significant for delays up to 300 ms.

Due to the increased nominal valve delay, an additional investigation of extended domain parameter limits and an evaluation for control frequencies of 5 and 10 Hz is performed. It could be shown that the reduced control frequency strongly increases the robustness to delays, however, the dynamic control performance is significantly affected. In addition, the evaluation showed for the DR that the maximum valve delay should be less than the control time step, otherwise the success of the training can be impaired. If larger delays have to be considered, an extended DR phase can be added or the action delay can be used.

Since the controller is designed for robustness to model uncertainties, a controller trained on the previous model can also be used successfully with the updated model. In order to ensure that the evaluation is not distorted, the previous trained controllers are also evaluated with the previous model in the next chapter.

Chapter 6

Controller Analysis

After the extensive study and analysis of DR, noise, system delays and hyperparameter tuning, this chapter deals with the analysis of the controller trained with the final configuration.
First, the final controller's design is presented before it is evaluated in the following sections. For the final controller, the training curriculum is designed in such a way that the controller is able to learn all capabilities to meet the requirements (Table 5.4) within the training domain and partially beyond, listed in Table C.2. The threshold value is a critical parameter and from the consideration between time effort and control performance an experimental determined value of -15 proves to be most suitable for this control task. After the initial training of the baseline solution, the automatic DR is activated. By modifying configuration parameters, it is possible to specify that the standard deviation is first increased in 0.05 steps up to 0.2 and then in 0.2 steps up to 2.0. After completion of the automatic DR phase with adaptive standard deviation, the training is continued for 40 training iterations which must reach the evaluation episode reward mean threshold. In this period, the training converges to a value of around -4 which is composed of the remaining steady-state error and settling times at set point changes. Then, Gaussian noise is applied for observations. After a maximum of 40 training iterations, actuator noise is also activated. After another up to 40 training iterations, first random and then constant observation delay is applied. The last two phases again comprise a maximum of 80 training iterations. Referring to the training configuration in Appendix C, one training iteration consists of 25000 time steps or with regard to the final controller with 10hz and an episode length of 88 seconds of 24 full control periods. Thus 40 training iterations correspond to one million time steps.
Looking at the TensorBoard diagram of the mean evaluation episode reward used for curriculum learning (Figure 6.1), it becomes clear that a maximum of 10 million time steps are required for training. Of these, the initial training of the baseline solution requires no more than 1.5 million time steps, the DR up to 4.5 million time steps, and the final noise and delay a maximum of 4 million time steps. Using a computer with 36 cores Intel®Xeon®Gold 6140 central processing unit, 128 Gigabyte random-access memory and a NVIDIA®GeForce®RTX 3090 graphics processing unit, one training with the settings from Appendix C takes almost 5 days. Considering the amount of trainings, including some that had to be restarted due to hardware issues, the time and effort required is quite impressive.

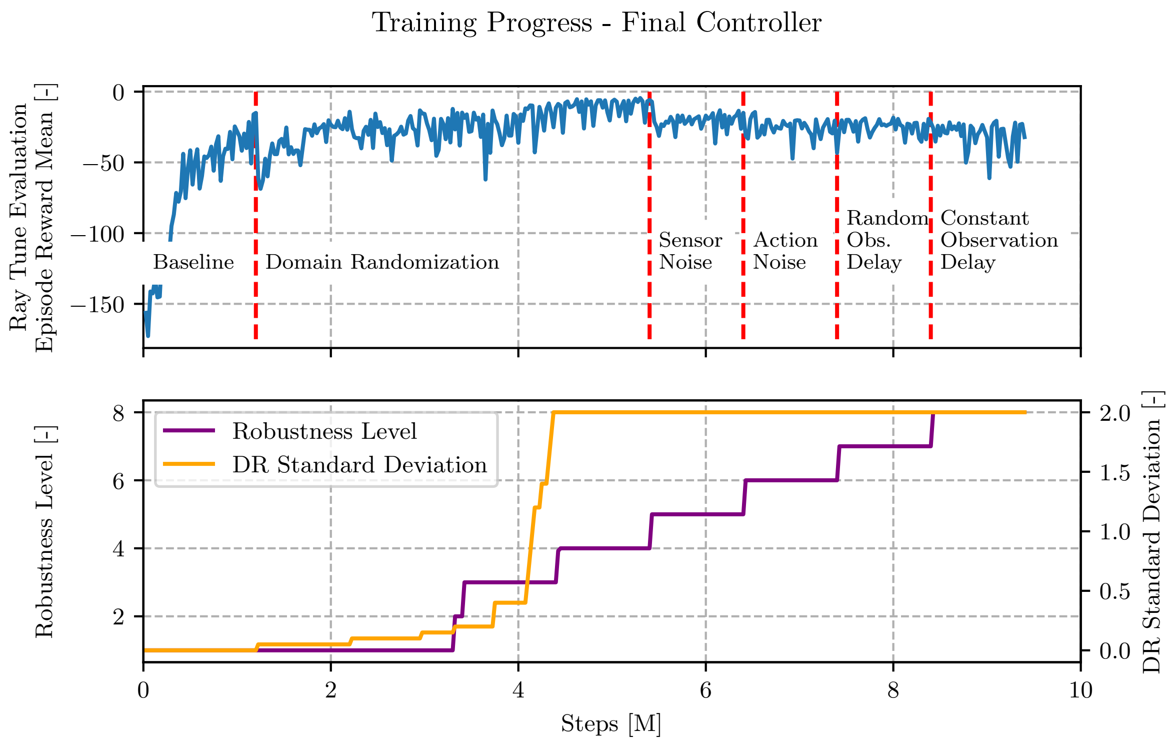


Figure 6.1: Final Controller Training Progress of Mean Evaluation Episode Reward with Data from TensorBoard

In preparation of the LUMEN FTP controller test campaign, additional controllers with reduced derated operating conditions are also trained and evaluated to be prepared in case of a modified test setup or changed environmental conditions. Derating is common practice in space projects to achieve higher reliability by reducing the operational requirements, thus increasing safety margins in critical operating areas. With respect to stability to dynamic perturbations such as noise and system delays, it can be beneficial to decrease the control frequency by accepting less control performance with regard to transient response behavior. In addition, the reference pump mass flow rate and discharge pressure can be changed to increase the stability margin with respect to operating constraints and the actuator saturation of both flow control valves. Besides the controller with 10 Hz, variants with a control frequency of 5 Hz and derated pump discharge pressures of 5 bar, 10 bar, and 20 bar, are prepared. The different reference trajectories are enclosed in Appendix A. In the following sections, the final controller operates at 10 Hz and is trained for a reference trajectory with 5 bar derated pump discharge pressure requirements.

6.1 Performance Evaluation

Within the performance evaluation, the focus is set on metrics measuring the transient response behavior as well as integral metrics to evaluate performance for the entire control trajectory. The control performance criteria used within this analysis are discussed in subsection 3.6.3 as

the reward is too abstract for the considerations of individual control aspects.
Besides the nominal case, the standard test cases for model mismatch will be a changed GN2 supply pressure provided by test bench P8.3 and an extended valve delay for the TFV flow control valve. As both parameters cause particularly large deviations, the following discussion will be more comprehensible due to the visible effects. Both parameters are closely related to pump discharge pressure and mass flow rate as the turbopump power is directly affected.
Moreover, different controllers are compared to demonstrate the superior control performance of this neural network based controller design compared to open loop control or partially trained controllers. The overall goal is the determination of the achievable control performance with respect to trajectory tracking. The disturbance rejection and performance deterioration are exemplarily discussed.

6.1.1 Trajectory Tracking

The main task of a controller is to adjust the actuators in such a way that the controlled variables follow a given reference by using the feedback from sensor measurements. The control objectives are stated with respect to the controller's ability to track the trajectory, which is the time-dependent definition of multiple set point changes.
Accordingly, transient response behavior criteria such as settling time, rise time, overshoot and steady-state error can be determined for each set point change whereby the step between the two operating point groups (t=49 s) is most relevant.
First, the valve sequence generated with the baseline controller (Figure 5.10) is used to evaluate the trajectory tracking performance for open loop control, shown in Figure 6.2.
In the nominal case, there is no reality gap due to different system characteristics or perturbations, so the designed valve sequence would be sufficient. With a settling time of less than two seconds and a steady state error of less than one percent, a high control accuracy is achieved. However, this ideal case does not occur in the real world. Model inaccuracies cannot be compensated due to the missing feedback signal from sensor measurements resulting in significant, unacceptable steady-state errors.
While open loop control valve sequences ensure a stable system behavior and prevent oscillations due to valve delays, this is a major concern in the controller design of closed loop controlled systems. Thus the control performance of the closed loop baseline controller is even worse for an extended valve delay. Evaluating Figures 6.2 and 6.3, there are significant oscillations excited by the closed loop baseline controller.
The oscillations of the controlled variables remain constant. Their amplitude depends on the valve delay and other model deviations such as turbopump efficiency, which changes the sensitivity to turbine power output changes. The baseline controller is not able to compensate this perturbation with appropriate valve commands and to damp the oscillations. As already discussed in Chapter 3 on the key concepts of RL, the baseline controller has only been trained for a specific control task, so that the policy approximated by the neural network has been specialized for training data specific properties in such a way that it cannot generalize to deviating scenarios. As expected, the control performance in the nominal case is identical for the baseline controller operated in open loop and closed loop.
By extending the training domain using DR, the robustness of the controller can be improved.

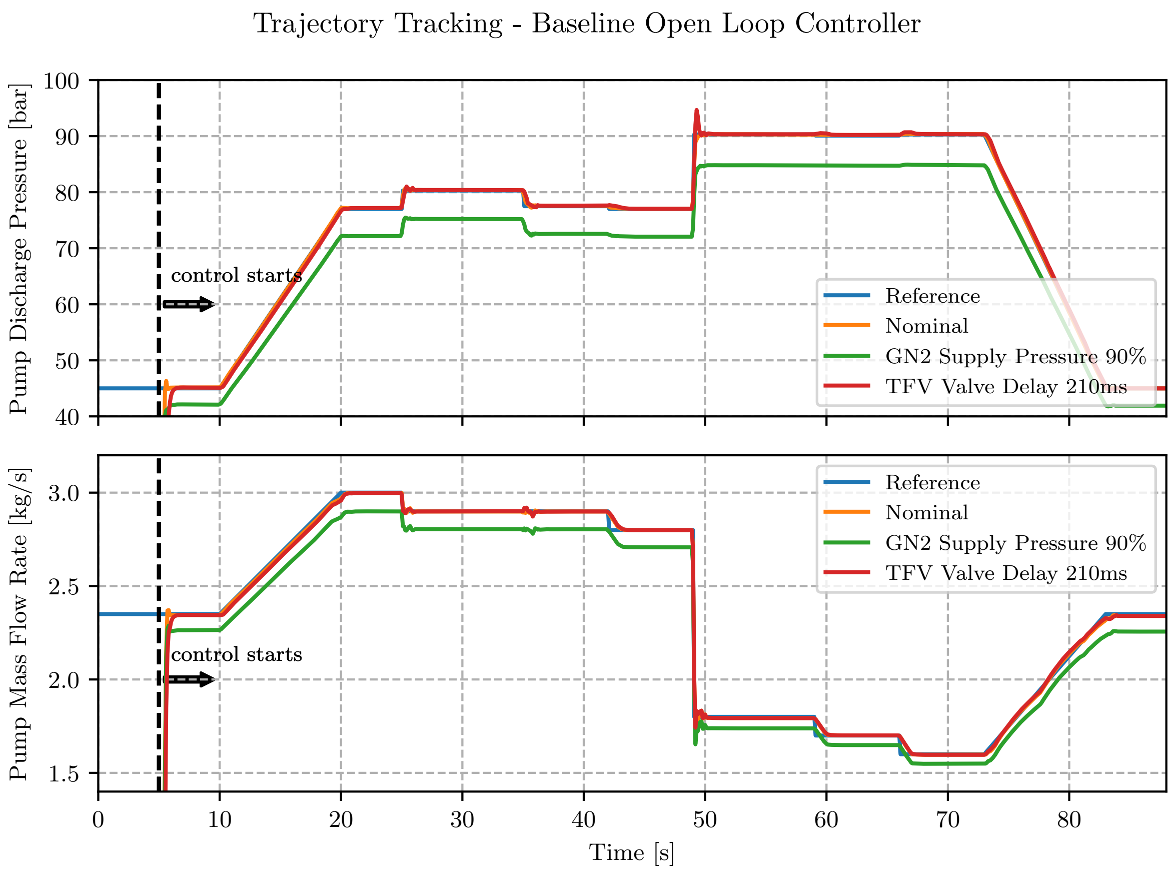


Figure 6.2: Baseline Open Loop Control

This is already obvious for the so-called single controller (Figure 6.4) where only the supply pressure for GN2 is varied in the range from 56 bar to 62 bar. The parameter is sampled per episode from a truncated normal distribution whose standard deviation is increased by 0.05 when a threshold of -15 is exceeded in the mean evaluation episode reward until 2.0 is reached. The offset error for a varied GN2 supply pressure is compensated so that the single controller achieves almost the same performance as for the nominal case. Remarkably, the disturbance is greater than what was included in training, in this example the difference between training and test is 6 bar. However, it is already clear that the sensitivity to valve delays has not changed but rather increased. As required, there is no deterioration of the control performance for the nominal case.
By extending the training domain with the parameters considered in the model sensitivity analysis, the training setup for the so-called complete controller is obtained. The DR parameters are listed in Table C.4 with nominal and boundary values. Since the parameter TEG_F.Ao was introduced only from the training series ID 05X on and the nominal value of the valve delay was also increased from 10 ms to 70 ms at the project end, the previously trained controllers are evaluated with the previous model.

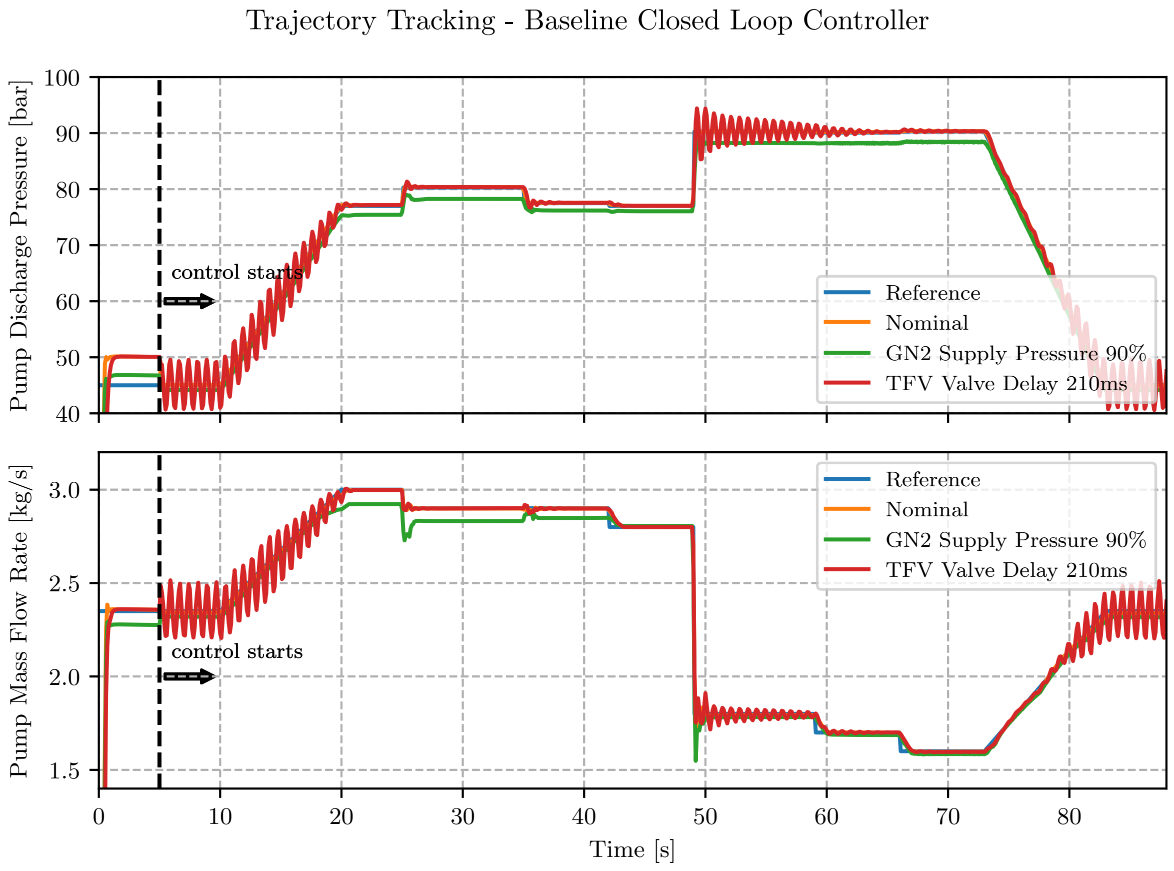


Figure 6.3: Baseline Closed Loop Control

The TEG_F.Ao component had to be updated to comply with the conditions and thus the efficiencies and performances achievable during the FTP controller test campaign. In particular the operation of the turbine with gaseous nitrogen with almost ambient temperature instead of the hot methane from the regenerative cooling cause large differences, which need to be considered by model modifications.
As the training domain expands, the controller becomes more robust, even with regard to valve delays. Introducing valve delay randomization of 0.005 s to 0.08 s with a nominal value of 0.01 s is already sufficient to impart a stabilizing oscillation damping mechanism into the control policy making it more robust to even larger delays like 210 ms, as shown in Figure 6.5.
The oscillations previously excited in the baseline and single controller are damped to such an extent that the control requirements are met. The steady-state error for a drop in GN2 test bench interface pressure is completely compensated.
Nevertheless, the DR did not cover noise and observation delay in training. If sensor noise is taken into account with a standard deviation of 2 % and sensor delay of one time step, at 10 Hz control frequency 100 ms, weaknesses of the complete controller become apparent. Due to the delayed observations, strong oscillations arise again, which cannot be compensated.

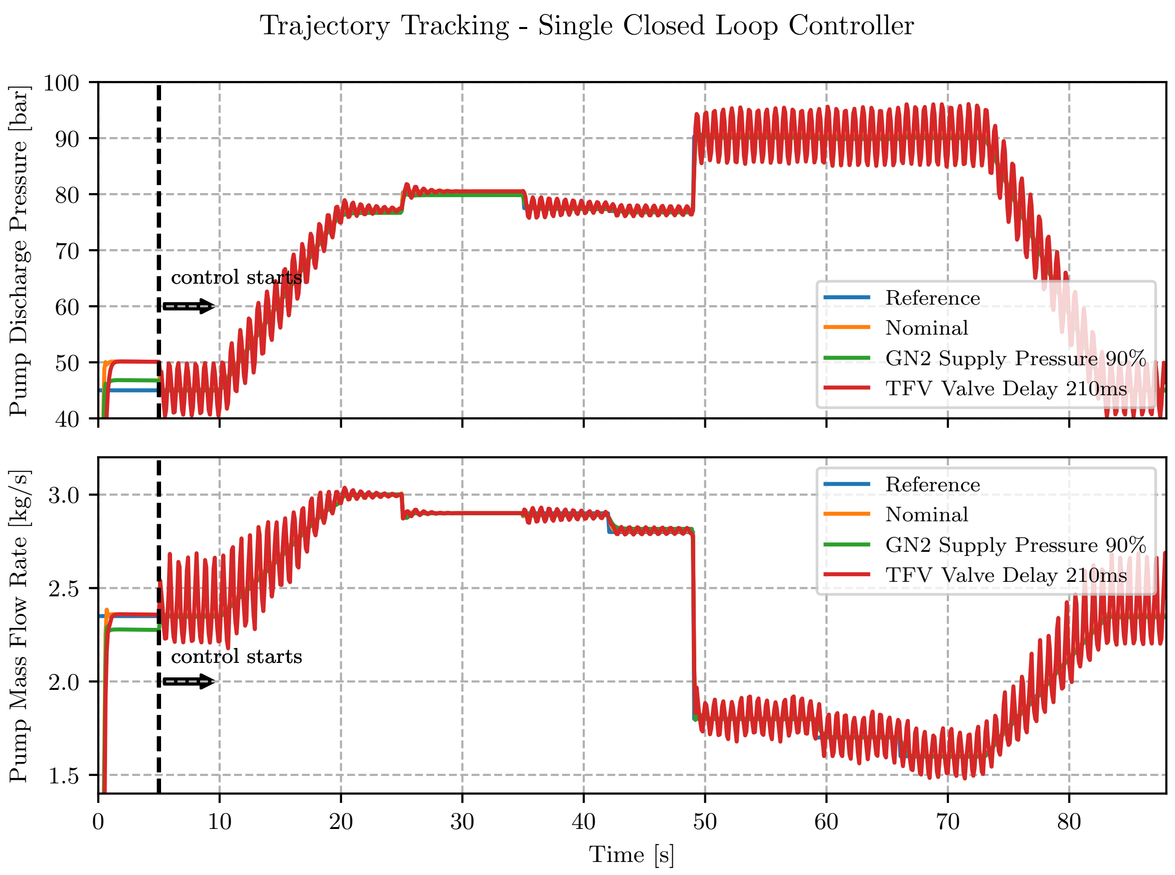


Figure 6.4: Closed Loop Control for Agent trainied with single DR parameter (GN2 Supply Pressure)

The sensor noise causes strong noise in the commanded valve positions, which even amplifies the noise of the controlled variables.
Additional introduced sensor noise and delay were not covered by the DR but have been added to the curriculum learning for the training of the final controller. Comparing complete and final controller, the necessity of such robust tools becomes evident (Figure 6.6 and 6.7).
Only the final controller is able to cope with all perturbations in compliance with all control requirements. This conclusion proves that AI-based methods need sufficient and diverse data during training to generalize to unseen test data different from the training domain. All these aspects take into account the real-world conditions at the test bench for the planned turbopump test and support the sim-to-real transfer. In the future, this approach can be extended to include sensor errors such as offset or linearity errors to enhance the robustness further. The neural network based controller can only be used if the sim-to-real transfer capabilities are guaranteed and validated by robustness analysis.
Nevertheless, it is also clear from the final controller that the control performance is degraded compared to the complete controller for the nominal case. This is mainly caused by the additional training levels but also by the model update.

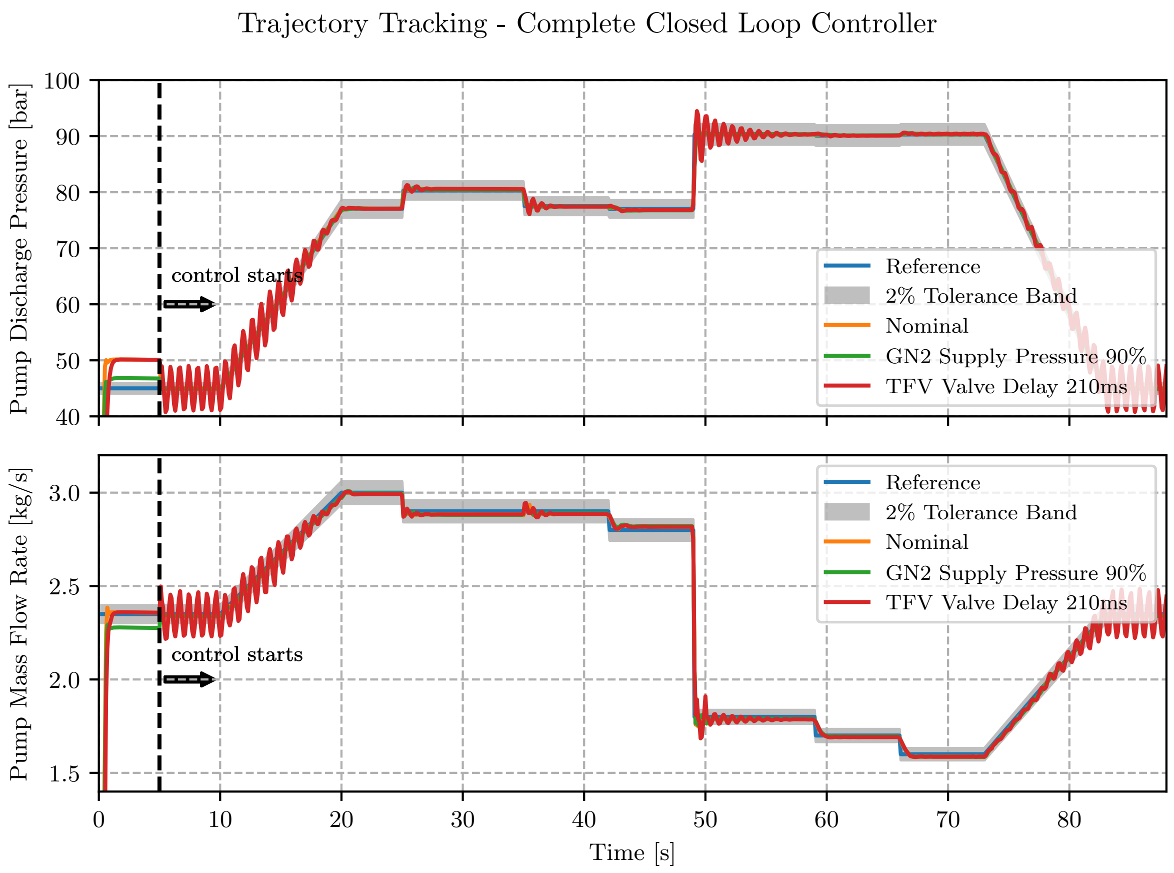


Figure 6.5: Complete Closed Loop Control

Several trainings evinced almost identical courses of the training and evaluation reward and related training progress with curriculum learning, so that a good stability and especially for real-world applications required reproducibility of the results could be proven. If the steady-state error should be reduced further, the tolerance band of the tracking penalty can be tightened accordingly. An appropriately configured controller confirmed the expected increase in training time by a factor of two. However, in the present work and considering the test campaign of several days, a shorter training duration was preferred.

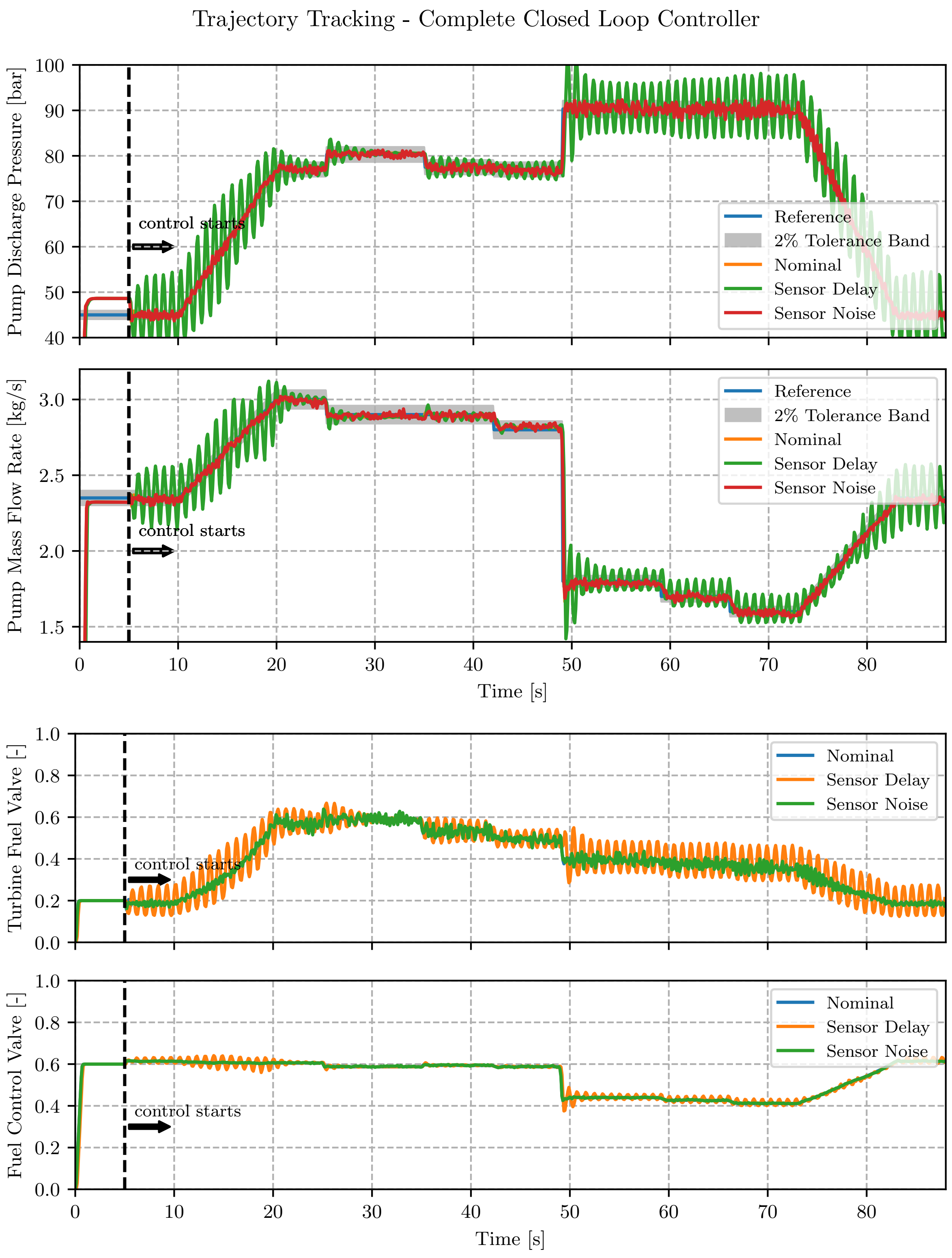


Figure 6.6: Complete Closed Loop Control with Sensor Delay and Noise

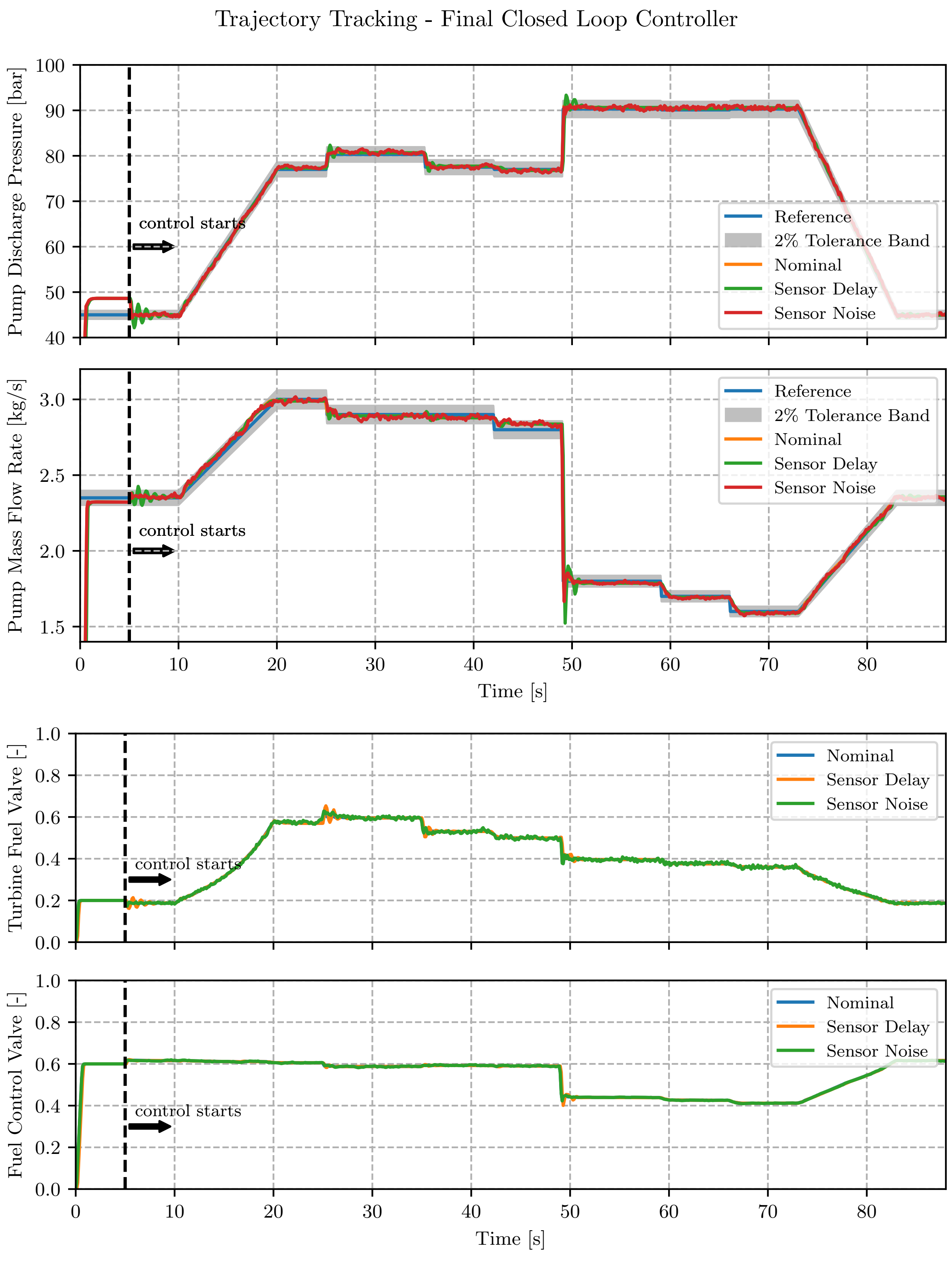


Figure 6.7: Final Closed Loop Control with Sensor Delay and Noise

6.1.2 Disturbance Rejection

Although the controller is designed with a focus on robustness improvements regarding model uncertainties, typical a major risk for neural network based controller trained only in simulation, the performance analysis is extended to investigate the disturbance rejection capabilities. Transient and static perturbations are applied at random times during the control trajectory representing failure cases as for instance a sticking valve or a leakage causing severe pressure drops. In particular, for a safety-critical and complex system like turbopumps or complete engines, the suppression of transient loads is difficult to accomplish but crucial to ensure safe operation and avoid abrasion and damage to extend the overall lifetime.
Here, the disturbance rejection capabilities should be shown exemplarily for a transient and static drop in testbench interface pressure of GN2 and LNG. Figure 6.8 shows the pressure of the pump fluid dropping sharply for a short time after 15 seconds and the supply pressure of the turbine dropping permanently by 25 % from the 45th second on. The first transient disturbance has almost no noticeable effect due to the fast controller reaction through the TFV valve, which provides additional turbopump power for this time. During the subsequent static pressure drop at the turbine inlet side, the controller also uses the TFV valve for compensation. However, the turbopump power is severely reduced so that it is impossible to provide the required pressure and hence increase the power again even if the TFV valve is fully opened. Or respectively, due to the physical limitation of the turbine pressure ratio, the turbopump power is reduced such that the required pump discharge pressure can no longer be maintained at the specified mass flow rate. The unavoidable deviations as a result of the physical conditions are so small that a safe shutdown can be guaranteed at any time. The valve position curve shows that the maximum opening is not exploited. This demonstrates that the neural network based controller was able to deduce that for a valve with a linear characteristic the pressure loss and mass flow rate in the upper position range would change negligibly little and the valve would only be moved unnecessarily if it had to be closed again later.

6.1.3 Performance Deterioration

Performance deterioration is particularly relevant for reusable components. Here, an exemplary extreme test case was examined in which the performance decreases linearly by 30 % from the 45 second until the episode ends. Although episodic time-dependent parameter variations have not been considered during training, instead only static changes within the context of DR, the controller is able to react extremely effectively to these changes. This is shown in Figure 6.9 since the control performance changes are negligible and the control requirements are still met. No peaks appear at the beginning of the performance degradation as the controller adapts rapidly to the altered system properties by continuously adjusting the TFV valve position even during normally stationary operation.
The controller achieved a remarkable performance with a MAPE of less than one percent. This exemplary study demonstrates the ability of RL methods combined with powerful tools for improved sim-to-real transfer to train a versatile high performance neural network based controller that combines robust, optimal and adaptive control.

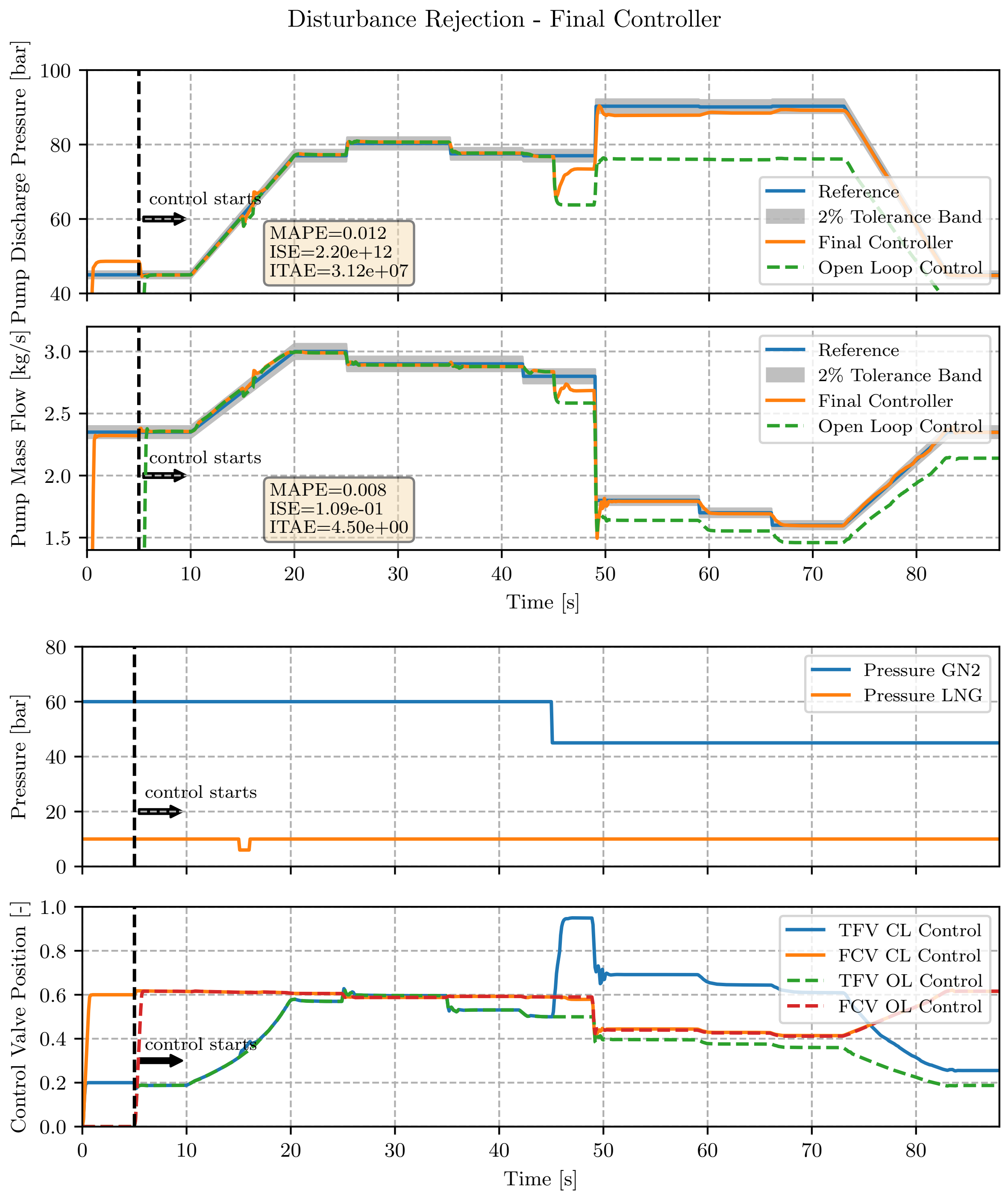


Figure 6.8: Disturbance Rejection for Final Controller

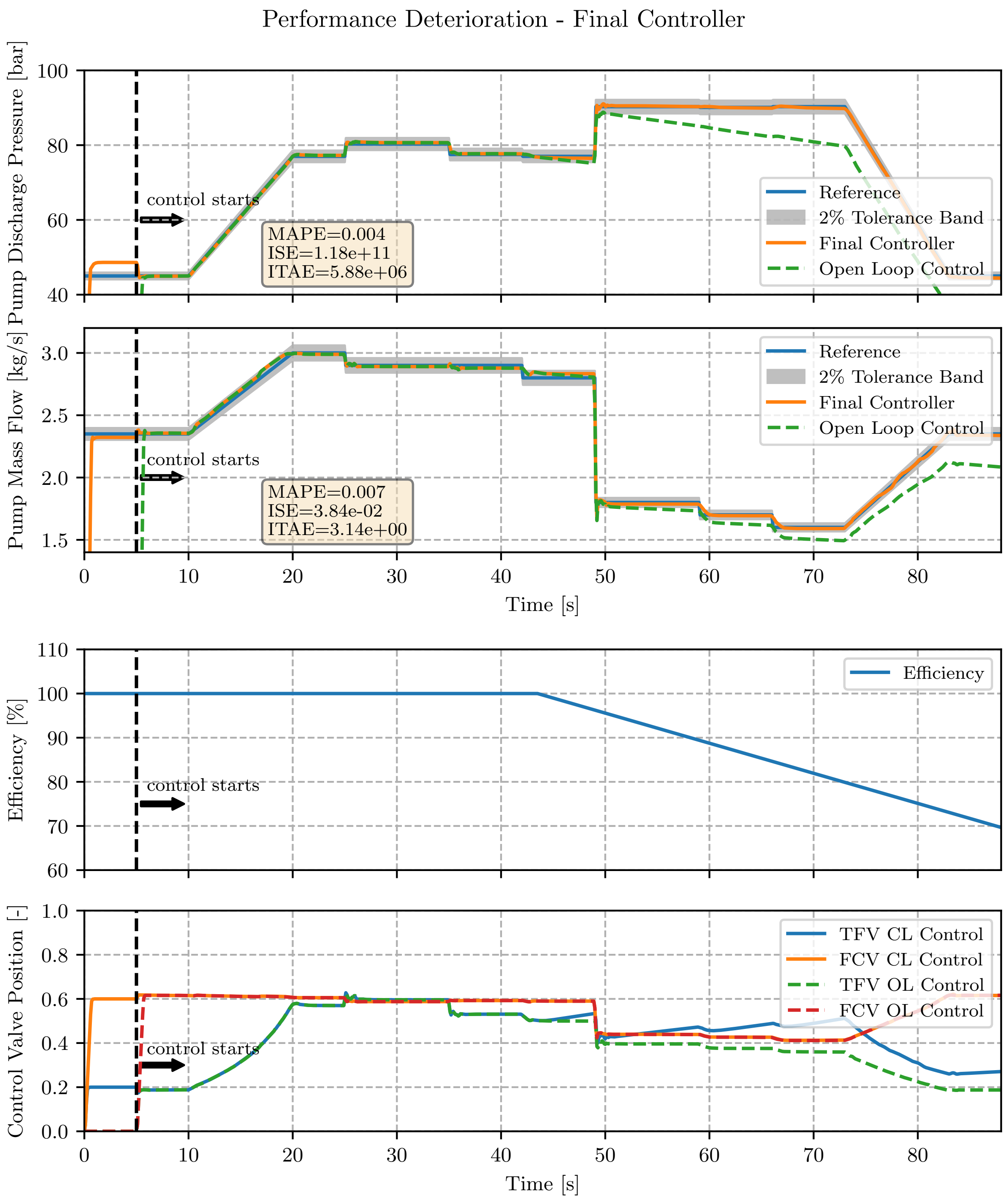


Figure 6.9: Performance Deterioration for Final Controller

6.2 Robustness Analysis

The robustness analysis is supposed to verify whether a certain performance can be achieved for a specific parameter space of model uncertainties. Moreover, the training dataset bounded by the DR limits is extended to untrained areas to measure the generalization capabilities of the control policy. While in classical robust control theory the control performance is optimized for the worst case, the neural network based controller is designed to achieve the control performance of the nominal case also for all other environments comparable to adaptive control. If the performance of the nominal case varies between different policies, it is mainly caused by the stochasticity of the training and the different total amount of training iterations.
The performance degradation curve of a control system can be described by a bathtub curve. The achievable performance is plotted for varying model parameters in both negative and positive direction from the nominal value. The exponential increase at the boundary regions corresponds to the worst case scenarios.
The sensitivity analysis comprises the evaluation of controllers, each trained with an individual domain parameter compared to baseline, complete and final controller. The Monte Carlo analysis extends this to a statistical performance evaluation of randomly sampled combinations of model parameter deviations for the final controller and the worst case analysis studies the final controller for worst model mismatch within the training domain. The results are used to specify the operating envelope of the neural network based control for the tests of the LUMEN FTP.

6.2.1 Sensitivity Analysis

The sensitivity analysis can also be considered as an analysis of the DR. This analysis is closely related to the model sensitivity analysis and investigates the robustness of the final controller with respect to model uncertainties. Similar to the model sensitivity analysis, the influence of a certain range of model uncertainties is evaluated for the control performance of both reference values. The model parameters considered in the sensitivity analysis, and covered by the domain parameter set, are evaluated. For the analysis of all trainings, matrix plots are generated. The varied parameters are shown on the x-axis and the individual trained RL agents on the y-axis. The colormaps visualize the control performance over the range of parameter variation where the red lines correspond to the DR boundaries. To smooth the gridding, a cubic interpolation is used. The analysis in this section comprises agents trained for the baseline, the single DR of GN2 supply pressure, the complete DR and the final controller case, shown in Figures 6.10, 6.11 and 6.12. Please find the complete evaluation in the Appendix D.
While the MAPE is a measure for the average control performance, the ISE rather weights overshoots and oscillations and the ITAE the steady-state behavior. In this work, the ITAE is calculated per operating point and summed up and thus does not include the ramps and start and end points of the reference trajectory.
In Figure 6.10, for the baseline controller, there are larger peripheral areas where the MAPE reaches a value of 5 % and more. However, for the nominal case shown in the center areas the controller achieves a good control performance of less than 2 %. In direct comparison with the controller trained for the supply pressure (p_GN2) with DR, the control performance for the nominal case improves and the outer areas of poorer performance are reduced for deviations of

the test bench interface conditions. Furthermore, the robustness also improves for deviations of other parameters, such as the efficiency (TP_Efficiency.eta), as the area of better performance enlarges and thus larger deviations can be tolerated. Considering the complete and final controller, where the complete DR parameter set was applied, a strong improvement of the robustness becomes evident. Only for large reductions of more than 25 % of the GN2 supply pressure and the turbopump efficiency, the control performance falls that the MAPE increases to more than 3 %. A similar behavior can be observed for ISE in Figure 6.11 and ITAE in Figure 6.12. It is interesting to note that for the complete and final controller the steady-state error for the pump mass flow rate, indicated by the ITAE, increases slightly, but nevertheless the MAPE remains within the specified control requirements.

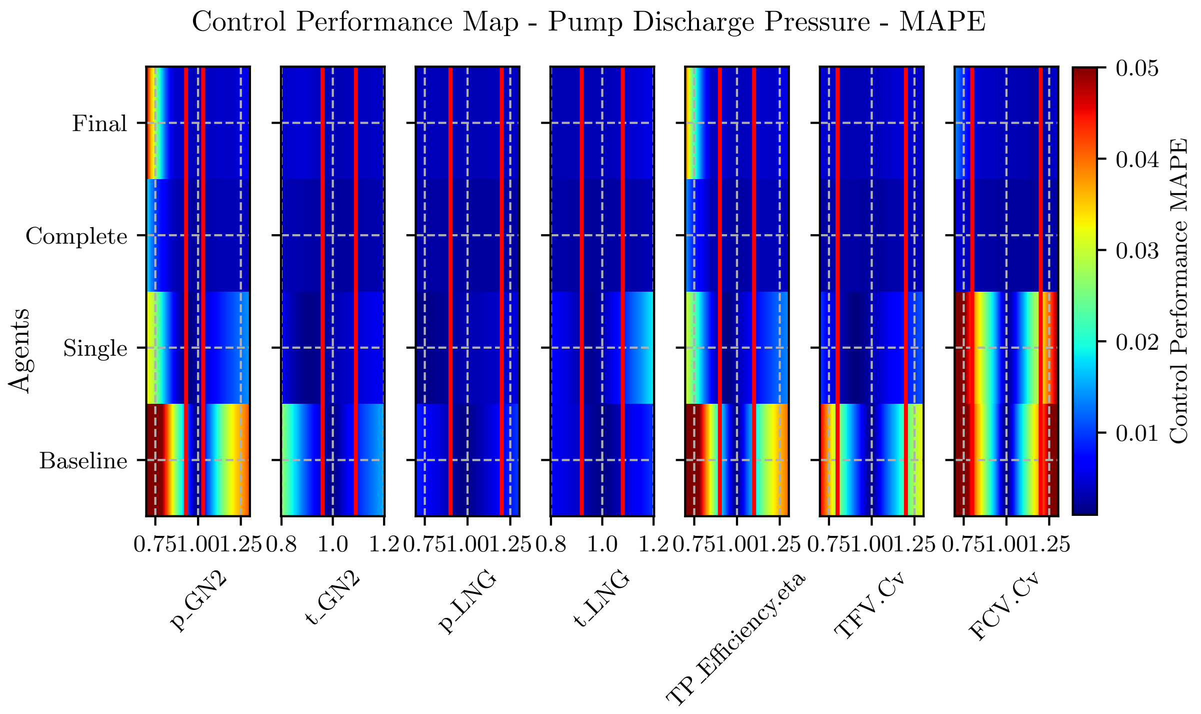
Considering the whole evaluation given in Appendix D, it is obvious that some parameters, including test bench interface pressures and temperatures as well as valve flow coefficients and turbopump efficiency, have a greater impact on the stationary behavior measured by ITAE than time-dependent valve parameters, turbopump inertia or system pressure losses. On the other hand, the ISE also shows robustness against valve delays, thus avoiding overshoots and oscillations. The turbopump inertia and pipe diameters affecting system pressure losses as well as the subsequently added parameter for the turbine exhaust gas nozzle area (TEG_F.Ao) have no remarkable effects due to the considerably lower order of magnitude. Furthermore, it could be shown that the DR of individual parameters not only complement each other but also improve together and achieve a very good performance and thus robustness within and beyond the training domain. The agents trained with the complete domain parameter set convinced in all areas in contrast to the baseline controller proving the successful application of Domain Randomization (DR) for the LUMEN FTP control task.

Moreover, it becomes evident that some domain parameters influence the robustness also for other model uncertainties. Here I have introduced the term of inherent robustness to describe the behavior that by carefully selecting a small set of domain parameters, the global robustness can be improved even for model uncertainties unknown to the control engineer. This concept is related to the definition of derived requirements in systems engineering if additional requirements implied but not stated in the subsystem specifications are necessary to describe processes and interfaces on system level.

In addition, small control performance changes for the nominal case are visible, which can be traced back to the stochasticity of trainings. Each training has an individual course due to the random environments that pose different challenges for the agent. In addition, there are also partly different training configurations depending on the available computing resources on different computers. For this reason, the analysis of individually trained agents is limited to significant differences because it actually requires a large number of agents to be able to draw statistical conclusions.

The main conclusion is that the evidence of successful DR for improving the robustness has been achieved. Based on the DR of individual parameters, it also becomes clear that each parameter contributes to the overall robustness in accordance with the results of the model sensitivity analysis. Furthermore, the DR of individual parameters also affects the robustness to other parameter variations. In addition, there is no case where the robustness to other parameter variations is significantly degraded.

A direct comparison of baseline and complete controller shows that the robustness of the individual parameters is not only combined but also improved by the joint training. The complete DR controller performs very well within the training domain but is also robust to larger deviations proving its generalization capabilities. Looking at the final controller, the same performance is achieved despite the additional training of noise and delay. From this it follows that the controller continues to learn while retaining previously acquired knowledge.



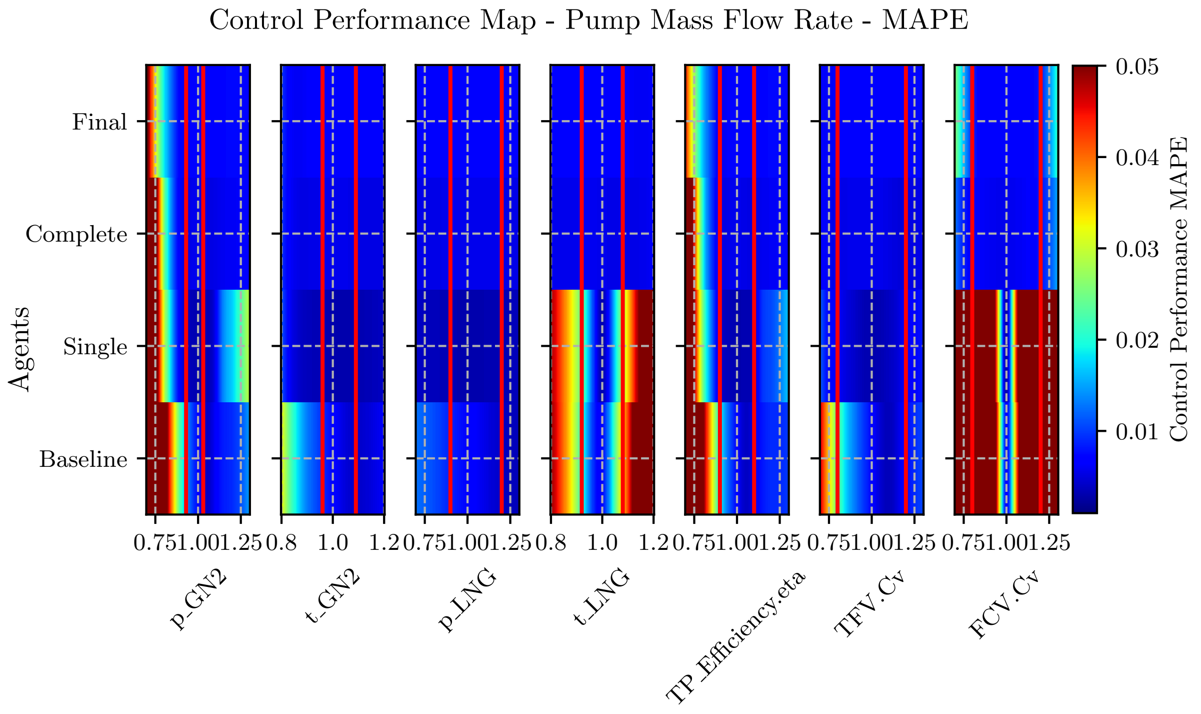
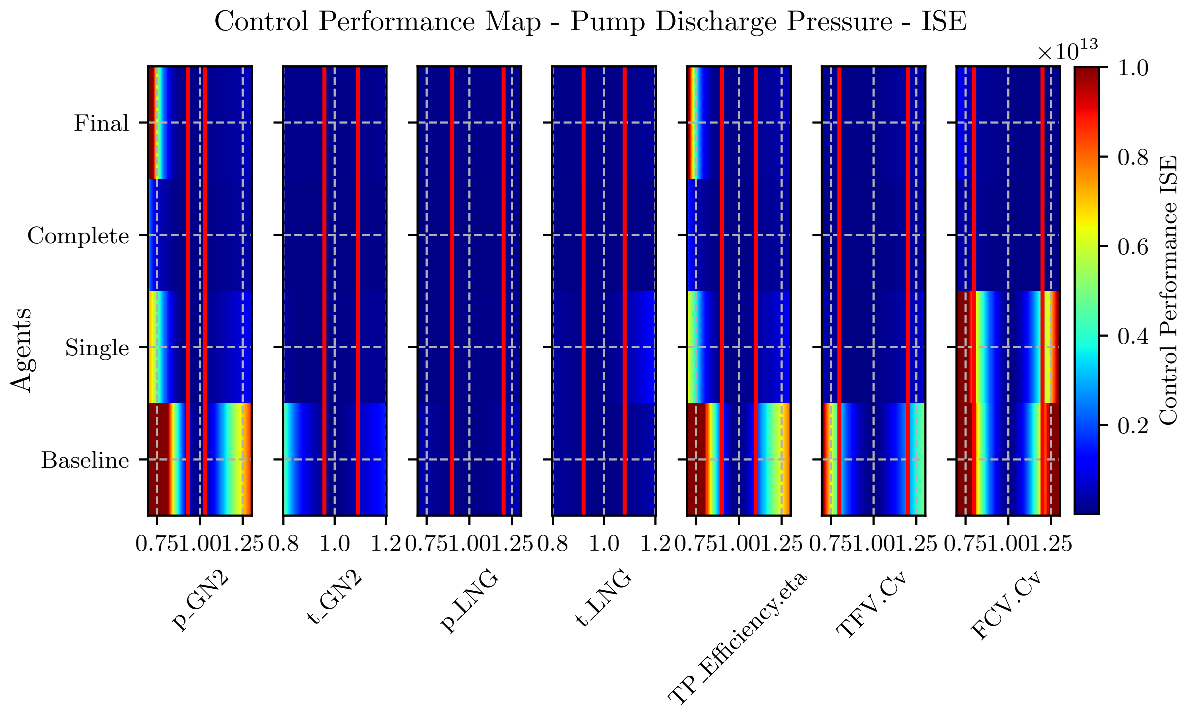


Figure 6.10: DR Analysis - MAPE



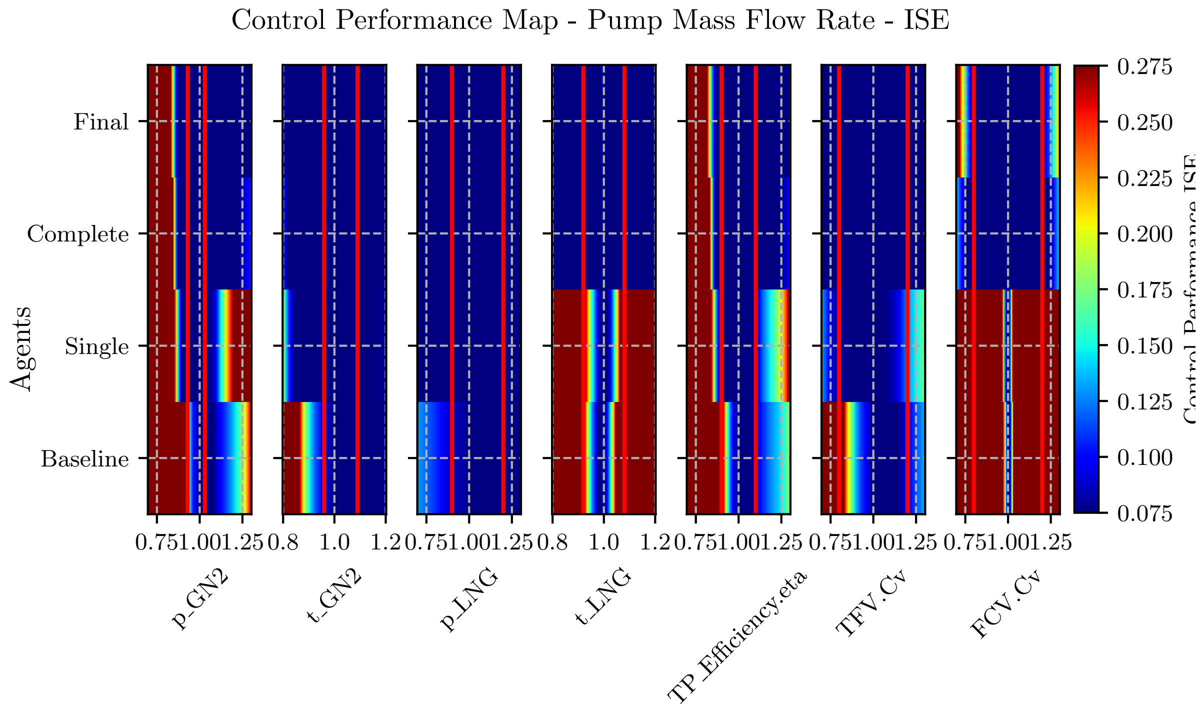
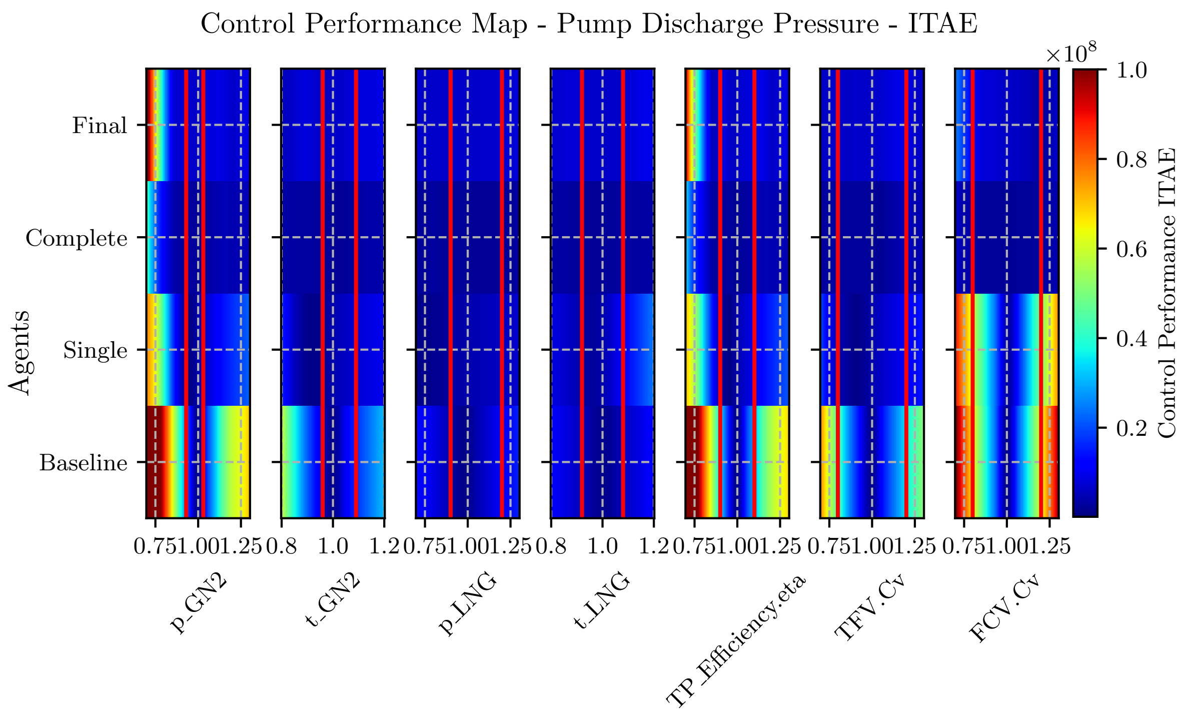


Figure 6.11: DR Analysis - ISE



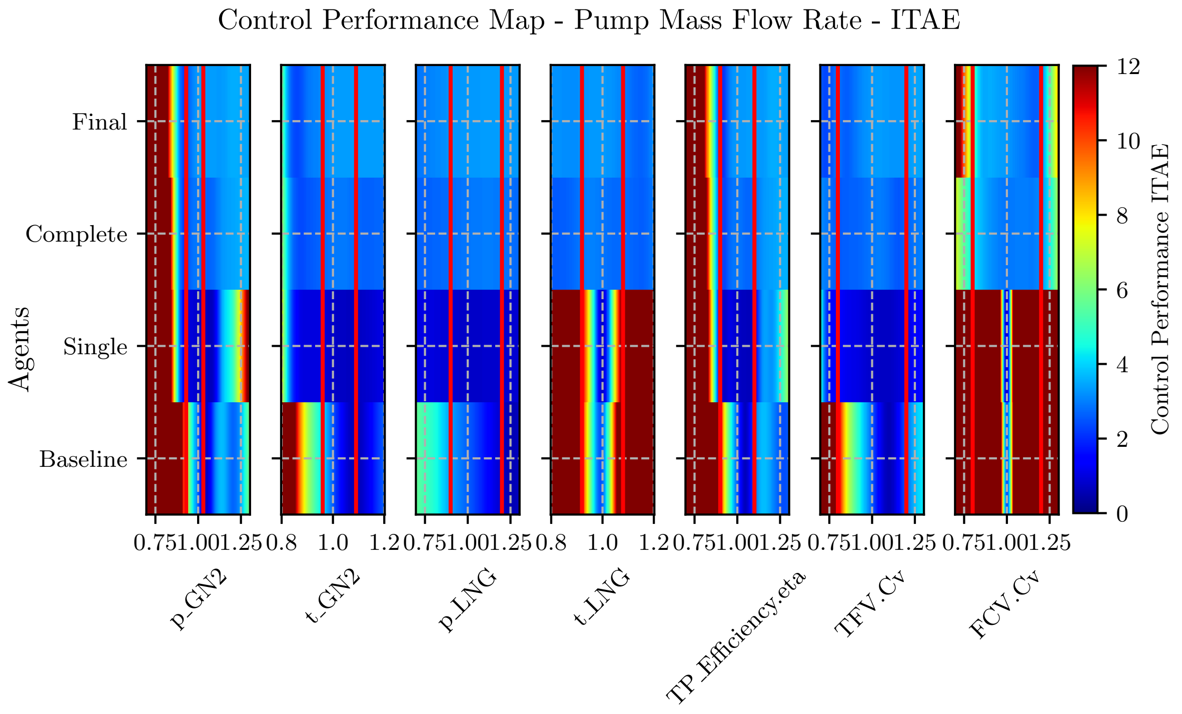
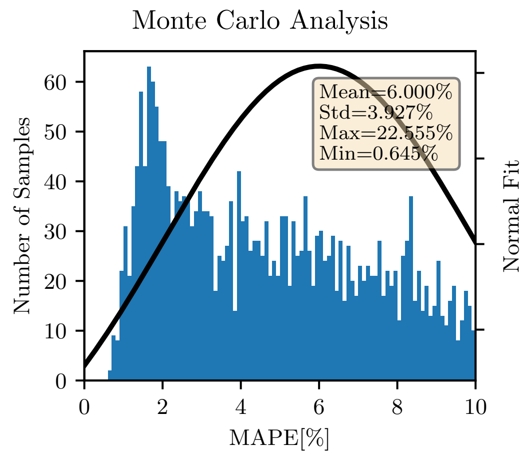


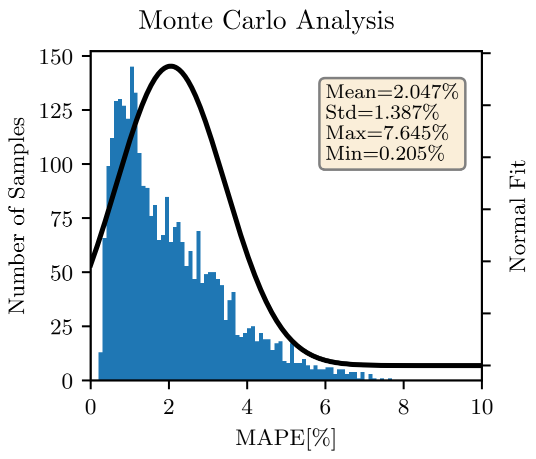
Figure 6.12: DR Analysis - ITAE

6.2.2 Monte Carlo Analysis

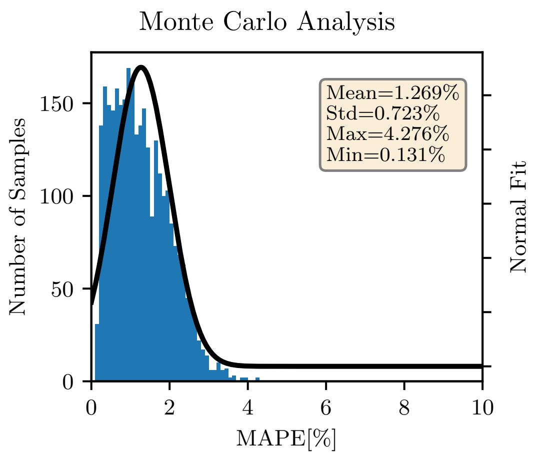
The Monte Carlo analysis allows the statistical evaluation of randomly chosen models with various parameter deviations. After using the agents trained on the previous model for the sensitivity analysis, the Monte Carlo analysis is performed with the updated model to show the robustness not only for individual parameter changes but for different combinations. With in total 17 domain parameters, with seven significant model parameters following the results of the model sensitivity analysis and DR analysis, this is a high dimensional problem. The Monte Carlo analysis is a great tool to gain more information with a relatively simple analysis setup. With 3000 samples the final controller design can be evaluated considering the DR limits used during training or extended boundaries as test dataset. Often conservative approaches are used where all parameters are varied in a common range. In this Monte Carlo analysis, the boundaries for parameter variations are set to $\pm 10\%$ thus ensuring physical possible states. Normally only one or two parameters are varied to facilitate the analysis using graphs or two dimensional colormaps. Here such analysis tools are not applicable. It is therefore necessary to focus on histograms and related statistical metrics such as mean, standard deviation or variance. Moreover the sample with minimum performance gives insights in the worst case performance and will be used for the worst case analysis in the following chapter.
The uniform probability distribution is used for sampling to enforce also the sampling in boundary areas. The High performance Universal applicable Lampoldshausen Cluster (Hochperformanter Universell einsetzbarer Lampoldshausen Cluster) (HULC) accelerated the analysis with six nodes of 64 cores. Since the cluster uses linux as operating system, some scripts had to be modified accordingly.
In Figure 6.13 and 6.14 the histograms for open loop, baseline, single DR, complete DR and final control are shown for each reference variable. It is obvious that the control performance improves as the mean shifts to smaller areas based on the MAPE plotted on the x-axis. Furthermore, the probability distribution is concentrated in the range of less than 2 % MAPE by a continuously decreasing standard deviation. Remarkable is the significant improvement of the single controller for the pump discharge pressure, since it depends mainly on the turbopump power directly affected by this parameter. With these results, the controller is qualified for use in the upcoming LUMEN FTP controller test campaign which has already been approved in the test preparation meeting.



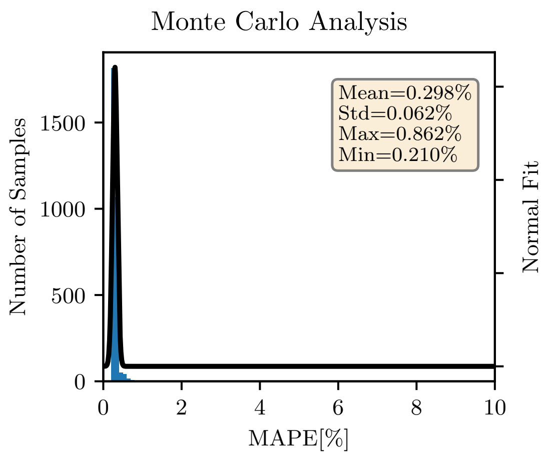
(a) Open Loop



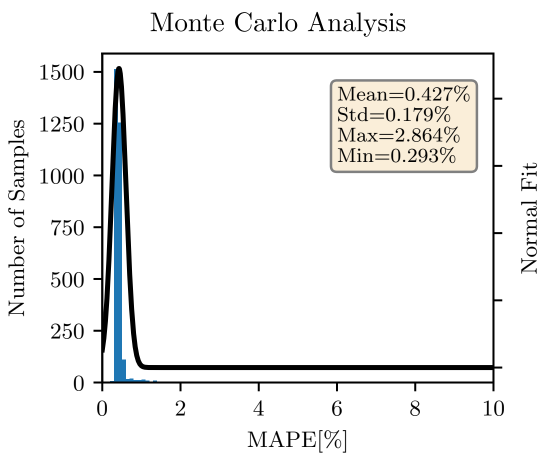
(b) Baseline



(c) Single DR

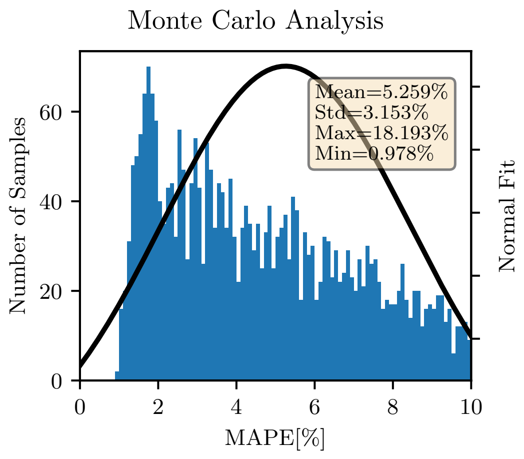


(d) Complete DR

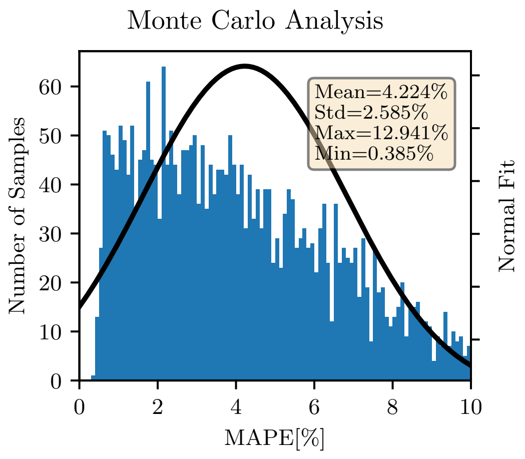


(e) Final

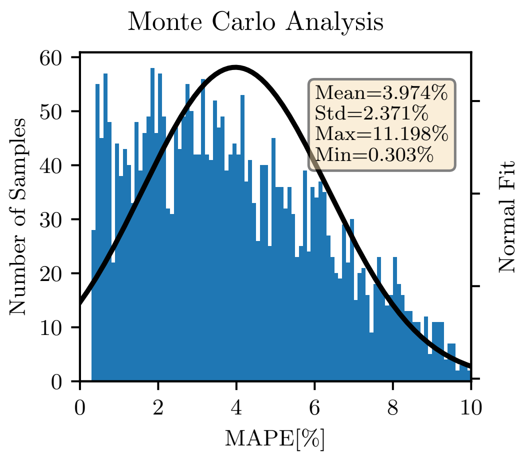
Figure 6.13: Monte Carlo Analysis for Pump Discharge Pressure and Model Variations of ±10%



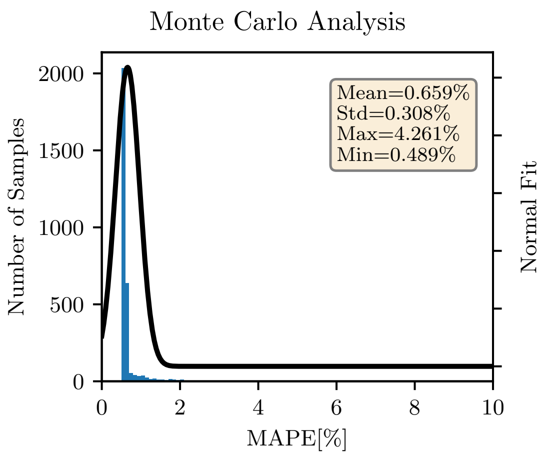
(a) Open Loop



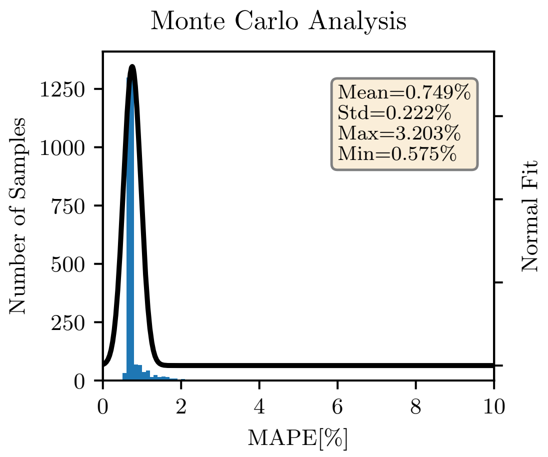
(b) Baseline



(c) Single DR



(d) Complete DR



(e) Final

Figure 6.14: Monte Carlo Analysis for Pump Mass Flow Rate and Model Variations of ±10%

6.2.3 Worst Case Analysis

The worst case analysis is a powerful tool to investigate the control performance for the most adverse model variation within or outside the training domain. This analysis can be further extended to model parameters and even physical relationships formulated by differential equations which are not considered during the robust design of the neural network based controller.
According to the Monte Carlo analysis samples with the worst performance, the change in turbopump efficiency and test bench supply pressure for the turbine working gas GN2 are most important for the control performance of the final controller. From the theoretical point of view, in addition to minimum turbopump efficiency and minimum supply pressure, increased turbopump inertia and increased system pressure losses result in an additional deterioration of the overall control performance.
In Figure 6.15, these parameters were accordingly adjusted by plus or minus 10 percent to represent the worst-case scenario. It is clear that here, the compensation can no longer be fully achieved using the flow control valves due to the operating conditions. In particular, the pressure drop on the turbine inlet side reduces the pressure ratio and the resulting power output to such an extent that the required pump discharge pressure and mass flow rate can no longer be maintained. At the test bench, such failures are detected by condition monitoring systems and, if a red line is endangered, a safe shutdown will be initiated. This evaluation demonstrates how robust this neural network based control is so that even large deviations are compensated that would not arise in real testing.

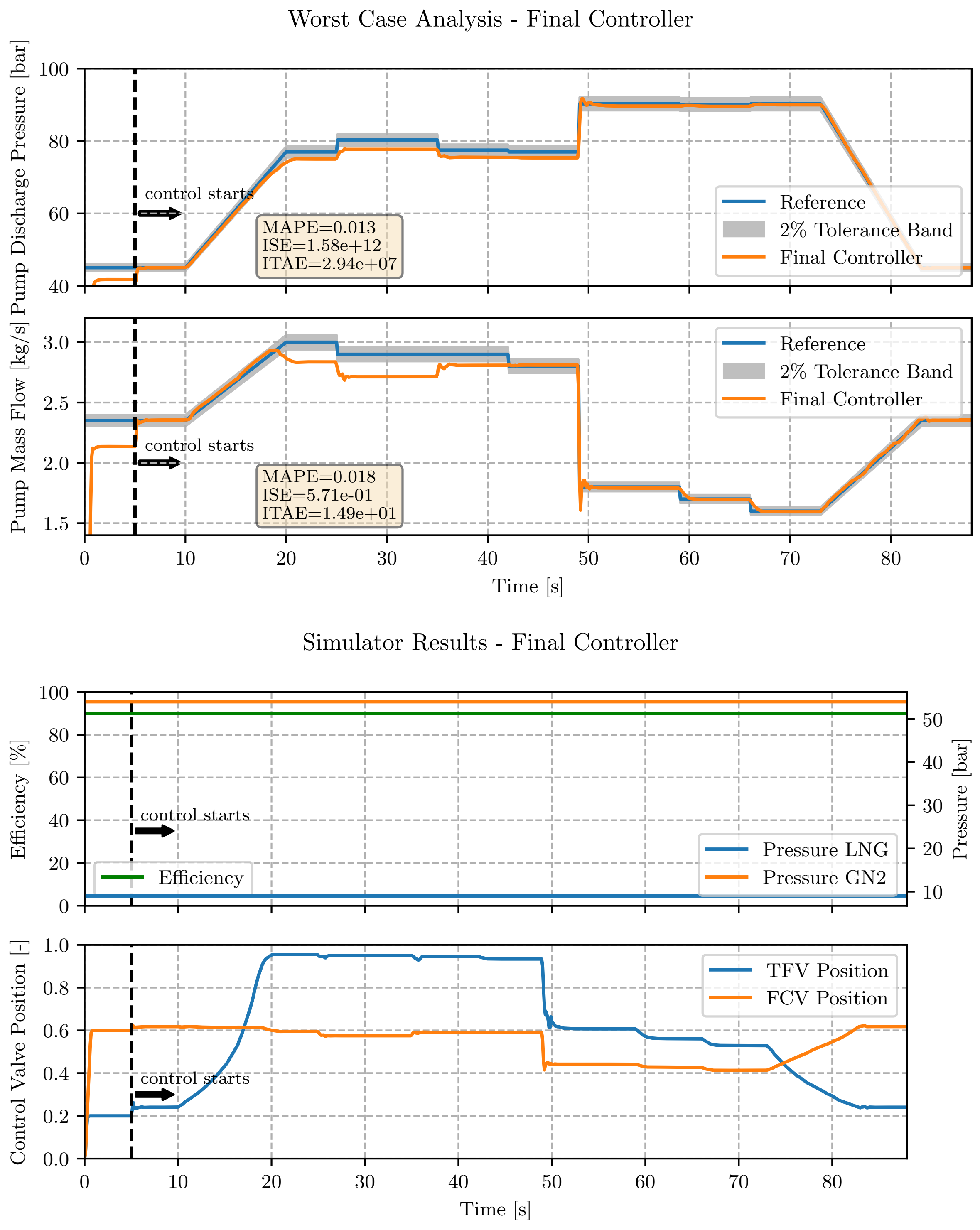


Figure 6.15: Worst Case Scenario for Final Controller

Chapter 7

Conclusion

Responding to increasing demands for advanced control methods for reusable rocket engines, this master thesis demonstrates the robust neural network based control for liquid rocket engine turbopumps using Reinforcement Learning (RL). Understanding the controlled system and its modeling is crucial to select the right methodology for a robust controller design. A comprehensive evaluation of individual design decisions assisted in gaining experience and knowledge in using different RL-related settings and robustness-enhancing tools to optimize the final controller for the specified control objectives.

Because of the absence of test results in preparation for the first tests, the controller was trained exclusively on a simulation model. For an application in the real world, the unavoidable model uncertainties and simplified assumptions regarding system delays and noise must be taken into account in the so-called sim-to-real transfer.

Curriculum learning and domain randomization allow continuous training with increasing difficulty due to additional perturbations of the nominal model. Reward function shaping and the use of future reference values and past observations give the RL agent the required capabilities to find an optimal control policy that can also react to varying environmental conditions through the learned system behavior.

The analysis of the controller configuration, the model parameter space and the statistical evaluations of the control performance enabled the identification of potential improvements, an understanding of the trade-offs of the RL agent and the training of a controller that achieves a high control performance combined with unmatched robustness. The developed controllers are ready for use at the test bench during the upcoming test campaign. This will be the first time at the DLR Institute of Space Propulsion that a neural network based controller is tested for a complex system in a real application out of simulation.

During the complete master project, it became clear that the success of a controller design depends on the collaboration of different engineering disciplines and requires systems engineering skills to optimize the overall system performance. Furthermore, software engineering and a good review culture ensure the code quality and integrity necessary to approach safety-critical rocket propulsion systems. The next milestone is clearly set on the intelligent engine control of LUMEN based on the preparatory work of this master thesis and the proof of concept of neural network based controller during the LUMEN FTP test campaign in autumn 2022.

Chapter 8

Future Work

From the development of a powerful yet versatile RL framework to the analysis of the most important design parameters and the robustness and performance analysis of the controller, the present work accomplished the necessary preparatory steps for an application during the upcoming test campaign. The most important goal, to prove the feasibility of a robust neural network-based controller in simulation, has been achieved. In preparation for the tests, however, there are still important aspects to consider, which will be briefly described here.
After the controller has been trained, it needs to be exported and deployed on an embedded system so that it can then be operated with the checkout system of the test facility. Due to the planned test sequence, it is already known that all operating points of the trained reference trajectory are performed before the controller is tested. This allows additional model updates and new findings to be considered in the modular, configurable training routine. In addition, a set of already trained controllers with different derating conditions assists in performing multiple tests per day and reacting rapidly to changes in the test setup or test conditions.
With regard to the control of the entire rocket propulsion system, the neural network-based controller should be evaluated in direct comparison with conventional PID and MPC control approaches. The current results are promising that Reinforcement Learning (RL) achieves a performance that is unattainable by classical control methods. However, the requirements of a safety-critical system such as a rocket engine also require further research in fault-tolerant and safe RL methods. Robust and optimal control support lifetime extension and enable throttling as well as restart capabilities required by future reusable rocket engines.
The LUMEN demonstrator project offers the next opportunity with a highly complex architecture, including six electrical flow control valves to show the power of AI-based control methods. Additional tools such as modified neural network structures using recurrent neural networks and the integration of target domain data could be key technologies for robust reinforcement learning to close the sim-to-real gap further. For the qualification and certification of modern machine learning concepts, the validation and verification of stability and safety requirements are crucial and should be a primary focus in future research activities. In preparation for the control of the LUMEN engine, the focus should be set in particular on safe and fault-tolerant control concepts.

Appendix A

Reference Trajectory

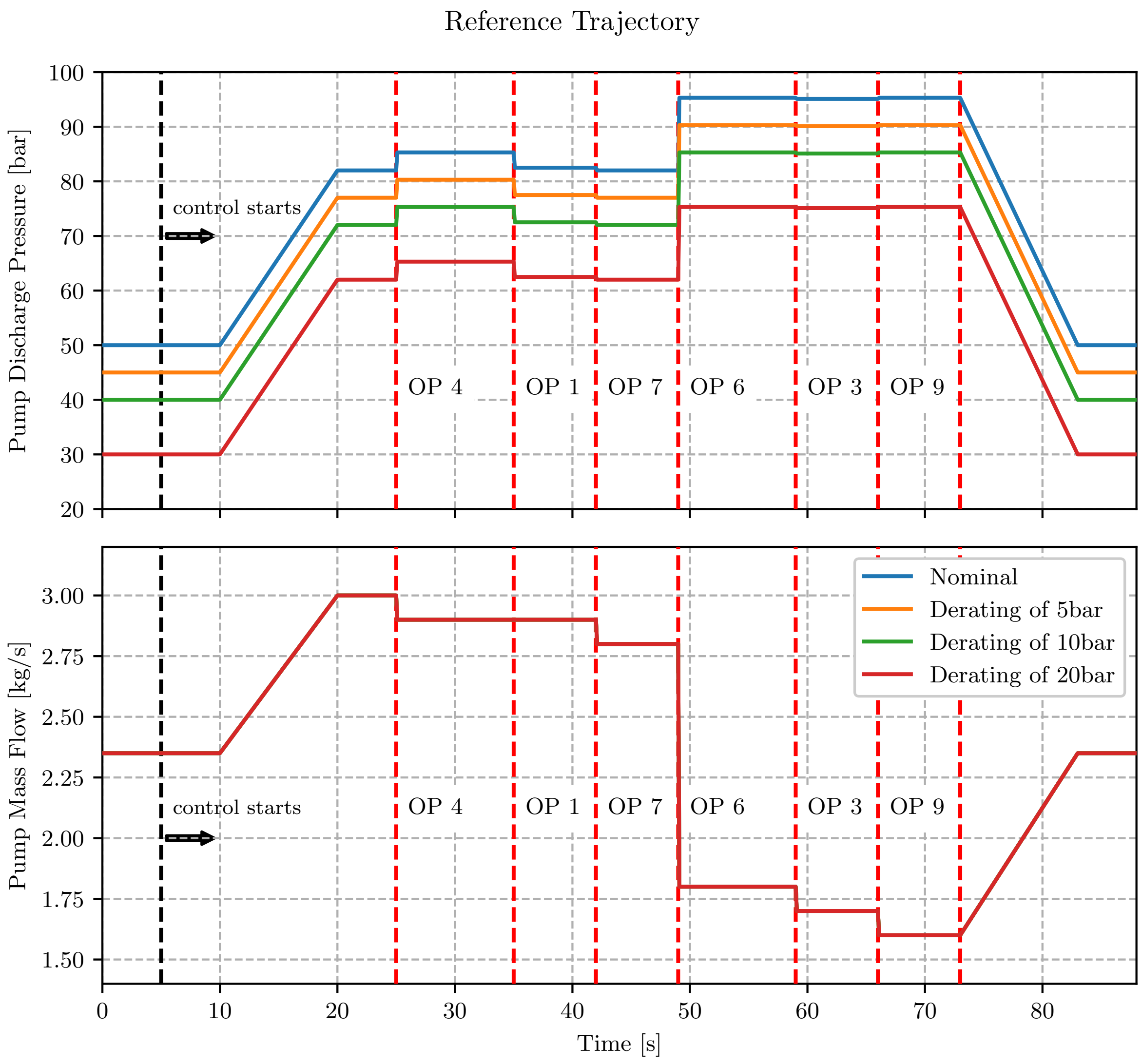


Figure A.1: Test Sequences for LUMEN FTP Controller Test Campaign

Appendix B

Reward Analysis

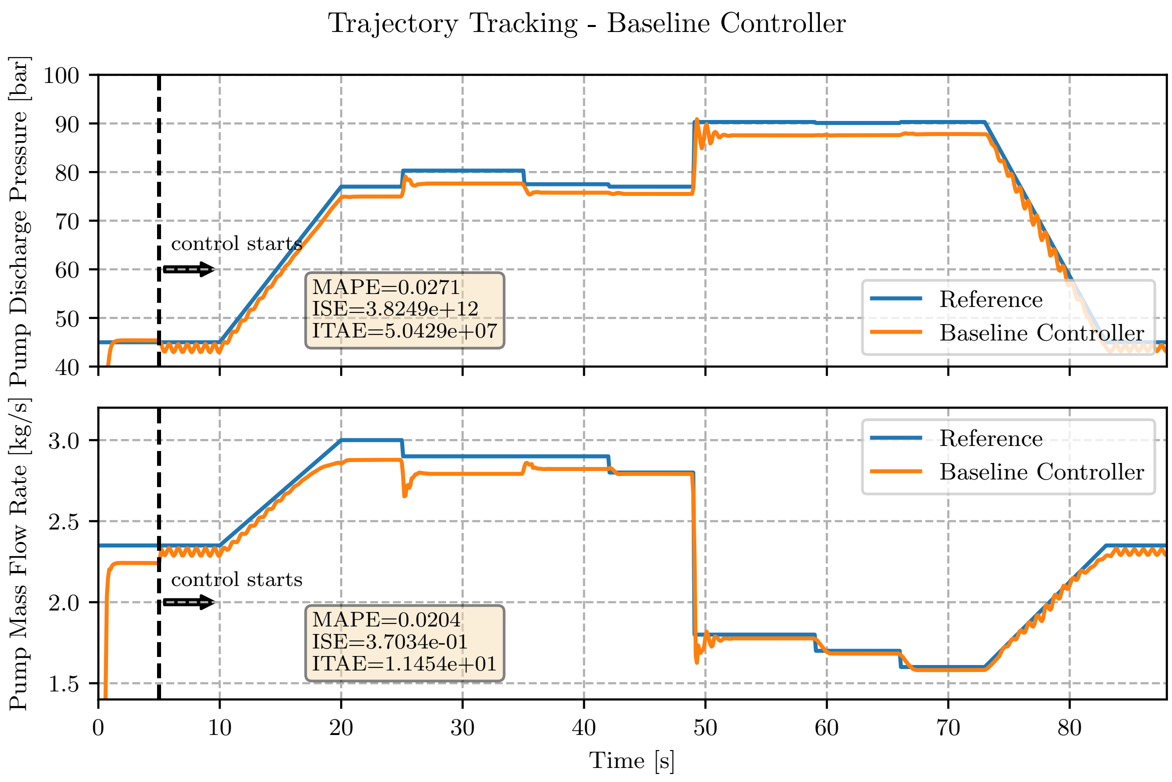


Figure B.1: Trajectory Tracking for Baseline Closed Loop Controller with 90% of Nominal Turbopump Efficiency and 210ms TFV valve delay

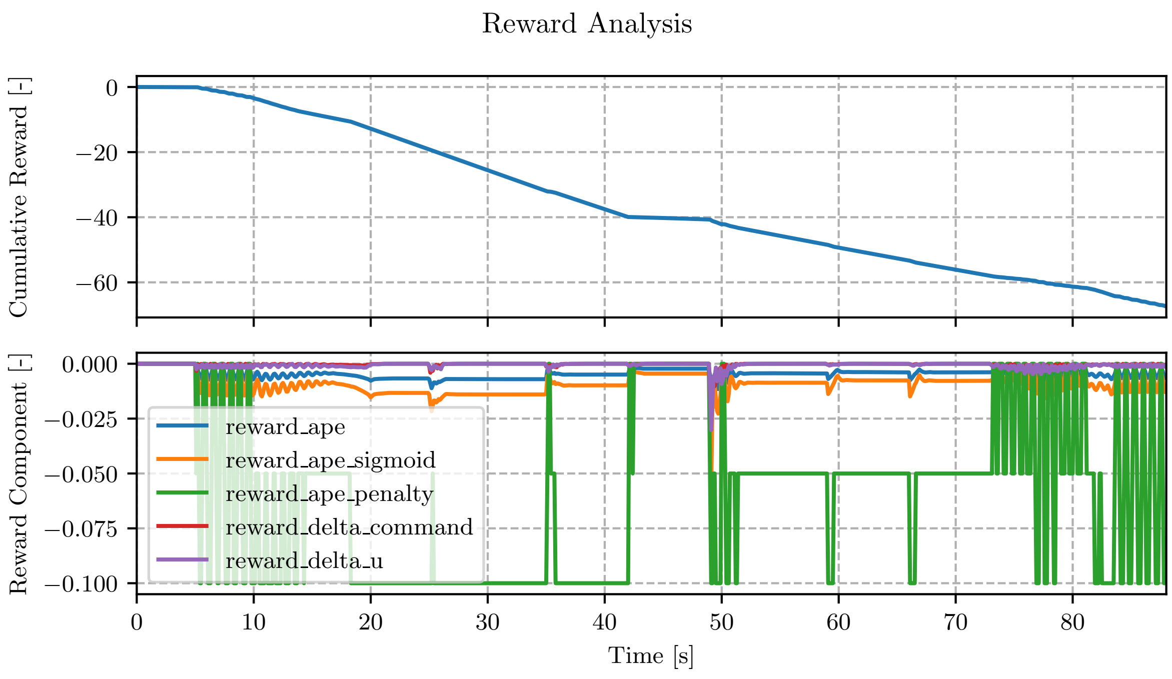


Figure B.2: Reward Analysis

Appendix C

Training

C.1 Training Configuration

Table C.1: DR Configuration

Parameter	Nominal	Minimum	Maximum	Unit
Delay_FCV.tdelay[1]	0.07	0.0	0.09	sec
Delay_TFV.tdelay[1]	0.07	0.0	0.09	sec
FCV.speed_max	1.0	0.8	1.2	1/sec
TFV.speed_max	1.0	0.8	1.2	1/sec
FCV.Cv	9.248	$0.8 \cdot 9.248$	$1.2 \cdot 9.248$	$\text{gpm/psi}^{0.5}$
TFV.Cv	9.248	$0.8 \cdot 9.248$	$1.2 \cdot 9.248$	$\text{gpm/psi}^{0.5}$
p_GN2	60e5	56e5	62e5	Pa
p_LNG	10e5	9e5	12e5	Pa
t_GN2	273.0	263.0	298.0	K
t_LNG	120.0	110.0	130.0	K
TP_Efficiency.eta	1.0	0.9	1.1	-
Pump_Fuel.I	0.001353	$0.9 \cdot 0.001353$	$1.1 \cdot 0.001353$	m^2
PipeFuel_03.D	0.02431	$0.7 \cdot 0.02431$	$1.3 \cdot 0.02431$	m
PipeFuel_04.D	0.02431	$0.7 \cdot 0.02431$	$1.3 \cdot 0.02431$	m
PipeFuel_16.D	0.02431	$0.7 \cdot 0.02431$	$1.3 \cdot 0.02431$	m
TEG_F.Ao	0.001335	$0.9 \cdot 0.001335$	$1.1 \cdot 0.001335$	m^2

Table C.2: Curriculum Learning Configuration

Level	Extend Training Scope	Set Parameter	Value
1	enModelRandomization	config.dr_std_dev_max	0.2
2	enModelRandomization	config.dr_std_dev_step	0.2
3	enModelRandomization	config.dr_std_dev_max	2.0
4	enModelRandomization	config.min_episodes_for_next_level	40
5	enGaussianSensorNoise	None	None
6	enGaussianActionNoise	None	None
7	enObservationDelay	enRandomDelay	1
8	enObservationDelay	enRandomDelay	0

Table C.3: RL Environment Configuration

Parameter	Value
TFV.min_pos and FCV.min_pos	0.1
TFV.max_pos and FCV.max_pos	1.0
TFV.init_pos	0.2
FCV.init_pos	0.6
time_step	0.1
time_control_starts	5
num_stacked_obs	10
num_future_ref	2
scale_factor_obs.p	1e5
scale_factor_obs.m	1
reward_config.tolerance_tracking_error	0.02
reward_config.penalty_tracking_error	0.5
reward_config.gain_tracking_error	2.0
reward_config.gain_delta_command	0.5
reward_config.gain_delta_u	0.5

Table C.4: DR Configuration

Parameter	Value
model_parameter	see Table C.1
parameter_deviations	see Table C.1
noisy_sensors	Pump_Fuel.f2.P
	Pump_Fuel.m
noisy_actuators	TFV_command
	FCV_command
sensor_noise_std_dev	0.02
actuator_noise_std_dev	0.01
observation_buffer_size	2
action_buffer_size	2
threshold_for_next_level	-15
max_episodes_for_next_level	40
min_episodes_for_next_level	0
dr_std_dev_step	0.05
dr_std_dev_max	1.0

Table C.5: Training and SAC Algorithm Configuration

Parameter	Value
framework	tensorflow
timesteps_per_iteration	25000
evaluation_interval	1
evaluation_duration	auto
evaluation_num_workers	3
evaluation_duration_unit	timesteps
evaluation_parallel_to_training	True
evaluation_config.explore	False
store_buffer_in_checkpoints	False
num_workers	5
num_gpus	1
sample_async	True
replay_buffer_config.capacity	1000000
gamma	0.90
optimization.actor_learning_rate	1.5e-4
optimization.critic_learning_rate	1.5e-4
optimization.entropy_learning_rate	1.5e-4
target_entropy	-3
metrics_num_episodes_for_smoothing	1
num_samples	1
checkpoint_freq	1
checkpoint_at_end	True

C.2 Training Plan

Table C.6: Training Plan (1/2)

ID	Trial Name	Date	DR	Model	Description
001_0	Trial_001_baseline_controller	20220630		1.0	baseline
001_1	Trial_001_baseline_controller	20220813		1.0	baseline
010_0	Trial_010_reward_actuator_penalty	20220607		1.0	gain_delta_command: 0.0
010_1	Trial_010_reward_actuator_penalty	20220607		1.0	gain_delta_command: 0.5
010_2	Trial_010_reward_actuator_penalty	20220607		1.0	gain_delta_command: 0.75
010_3	Trial_010_reward_actuator_penalty	20220607		1.0	gain_delta_command: 1.0
011_0	Trial_011_reward_control_penalty	20220615		1.0	gain_delta_u: 0.0
011_1	Trial_011_reward_control_penalty	20220615		1.0	gain_delta_u: 0.5
011_2	Trial_011_reward_control_penalty	20220615		1.0	gain_delta_u: 0.75
011_3	Trial_011_reward_control_penalty	20220615		1.0	gain_delta_u: 1.0
012_0	Trial_012_reward_tracking_error_penalty	20220609		1.0	tolerance_tracking_error: 0.02 / penalty_tracking_error: 0.5
012_1	Trial_012_reward_tracking_error_penalty	20220609		1.0	tolerance_tracking_error: 0.02 / penalty_tracking_error: 1.0
012_2	Trial_012_reward_tracking_error_penalty	20220609		1.0	tolerance_tracking_error: 0.01 / penalty_tracking_error: 0.5
012_3	Trial_012_reward_tracking_error_penalty	20220609		1.0	tolerance_tracking_error: 0.01 / penalty_tracking_error: 1.0
012_4	Trial_012_reward_tracking_error_penalty	20220609		1.0	tolerance_tracking_error: 0.01 / penalty_tracking_error: 0.0
012b_0	Trial_012b_reward_tracking_error_penalty	20220614		1.0	tolerance_tracking_error: 0.01 / penalty_tracking_error: 0.0 / threshold: -10
012b_1	Trial_012b_reward_tracking_error_penalty	20220614		1.0	tolerance_tracking_error: 0.01 / penalty_tracking_error: 0.5 / threshold: -15
012b_2	Trial_012b_reward_tracking_error_penalty	20220614		1.0	tolerance_tracking_error: 0.02 / penalty_tracking_error: 0.5 / threshold: -15
013_0	Trial_013_control_frequency	20220609		1.0	time_step: 0.1
013_1	Trial_013_control_frequency	20220609		1.0	time_step: 0.2
013_2	Trial_013_control_frequency	20220609		1.0	time_step: 0.5
013_3	Trial_013_control_frequency	20220609		1.0	time_step: 1.0
014_0	Trial_014_num_future_ref	20220607		1.0	num_future_ref: 1
014_1	Trial_014_num_future_ref	20220607		1.0	num_future_ref: 5
014_2	Trial_014_num_future_ref	20220607		1.0	num_future_ref: 10
015_0	Trial_015_num_stacked_obs	20220621		1.0	num_stacked_obs: 1
015_1	Trial_015_num_stacked_obs	20220621		1.0	num_stacked_obs: 5
015_2	Trial_015_num_stacked_obs	20220621		1.0	num_stacked_obs: 10
020_0	Trial_020_turbine_interface	20220622		1.0	p_GN2
020_1	Trial_020_turbine_interface	20220622		1.0	t_GN2
021_0	Trial_021_pump_interface	20220622		1.0	p_LNG
021_1	Trial_021_pump_interface	20220622		1.0	t_LNG
022_0	Trial_022_valve_delay	20220624		1.0	Delay_TFV.tdelay[1]
022_1	Trial_022_valve_delay	20220624		1.0	Delay_FCV.tdelay[1]
023_0	Trial_023_valve_speed	20220624		1.0	TFV.speed_max
023_1	Trial_023_valve_speed	20220624		1.0	FCV.speed_max
024_0	Trial_024_valve_flow_coefficient	20220627		1.0	TFV.Cv
024_1	Trial_024_valve_flow_coefficient	20220627		1.0	FCV.Cv
025_0	Trial_025_turbopump_efficiency	20220705		1.0	TP_Efficiency.eta
026_0	Trial_026_turbopump_inertia	20220627		1.0	Pump_Fuel.I
027_0	Trial_027_system_pressure_losses	20220627		1.0	PipeFuel_03.D
027_1	Trial_027_system_pressure_losses	20220627		1.0	PipeFuel_04.D
027_2	Trial_027_system_pressure_losses	20220627		1.0	PipeFuel_16.D
028_2	Trial_028_complete	20220707	X	1.0	complete
030_0	Trial_030_sensor_noise	20220721	X	1.0	enGaussianSensorNoise
030_1	Trial_030_sensor_noise	20220721	X	1.0	enOrnsteinUhlenbeckSensorNoise
031_0	Trial_031_action_noise	20220721	X	1.0	enGaussianActionNoise
031_1	Trial_031_action_noise	20220721	X	1.0	enOrnsteinUhlenbeckActionNoise
032_0	Trial_032_observation_delay	20220728	X	1.0	enObservationDelay
032_1	Trial_032_observation_delay	20220728	X	1.0	enObservationDelay / enRandomDelay
033_0	Trial_033_action_delay	20220730	X	1.0	enActionDelay
033_1	Trial_033_action_delay	20220730	X	1.0	enActionDelay / enRandomDelay
040_0	Trial_040_num_future_ref	20220711	X	1.0	num_future_ref: 1
040_1	Trial_040_num_future_ref	20220711	X	1.0	num_future_ref: 5
040_2	Trial_040_num_future_ref	20220711	X	1.0	num_future_ref: 10
041_0	Trial_041_num_stacked_obs	20220718	X	1.0	num_stacked_obs: 1
041_1	Trial_041_num_stacked_obs	20220718	X	1.0	num_stacked_obs: 5
041_2	Trial_041_num_stacked_obs	20220718	X	1.0	num_stacked_obs: 10
042_0	Trial_042_extended_obs_space	20220715	X	1.0	extended_obs_space with p_GN2, p_LNG, t_GN2, t_LNG
043_0	Trial_043_action_in_obs_space	20220715	X	1.0	action_in_obs_space without valve commands
044_0	Trial_044_currriculum_learning	20220707	X	1.0	curriculum std_dev_step 0.02
044_1	Trial_044_currriculum_learning	20220707	X	1.0	curriculum std_dev_step 0.1
044_2	Trial_044_currriculum_learning	20220707	X	1.0	curriculum adaptive std_dev_step 0.02 until 0.2 then 0.2

Table C.7: Training Plan (2/2)

ID	Trial Name	Date	DR	Model	Description
050_0	Trial_050_extended_valve_delay_10Hz	20220730	X	2.0	extended TFV valve delay 10Hz
050_1	Trial_050_extended_valve_delay_10Hz	20220730	X	2.0	extended FCV valve delay 10Hz
051_0	Trial_051_extended_valve_delay_5Hz	20220730	X	2.0	extended TFV valve delay 5Hz
051_1	Trial_051_extended_valve_delay_5Hz	20220730	X	2.0	extended FCV valve delay 5Hz
052_0	Trial_052_turbine_outlet_junction	20220730	X	2.0	TEG_F.Ao 10Hz
053_0	Trial_053_complete_10Hz	20220801	X	2.0	complete with extended valve delays and TEG_F.Ao 10Hz
054_0	Trial_054_complete_5Hz	20220803	X	2.0	complete with extended valve delays and TEG_F.Ao 5Hz
055_0	Trial_055_complete	20220815	X	2.0	complete with TEG_F.Ao
100_0	Trial_100_final_controller	20220726	X	1.0	final controller without delay (gain_delta_command: 1.5, gain_delt_u:1.5, num_future_ref: 5)
100_1	Trial_100_final_controller	20220803	X	1.0	final controller (gain_delta_command: 1.5, gain_delt_u:1.5, num_future_ref: 5)
101_0	Trial_101_final_controller	20220808	X	2.0	final controller (gain_delta_command: 1.5, gain_delt_u:1.5, num_future_ref: 5)
102_0	Trial_102_final_controller	20220808	X	2.0	final controller (gain_delta_command: 1.5, gain_delt_u:1.5, num_future_ref: 5)
103_0	Trial_103_final_controller	20220811	X	2.0	final controller without delay (gain_delta_command: 1.5, gain_delt_u:1.5, num_future_ref: 5)
104_0	Trial_104_final_controller	20220814	X	2.0	final controller (gain_delta_command: 1.5, gain_delt_u:1.5)
105_0	Trial_105_final_controller	20220817	X	2.0	final controller (gain_delta_command: 1.5, gain_delt_u:1.5, tolerance_tracking_error: 0.01)
105_1	Trial_105_final_controller	20220817	X	2.0	final controller

Appendix D

Domain Randomization Analysis

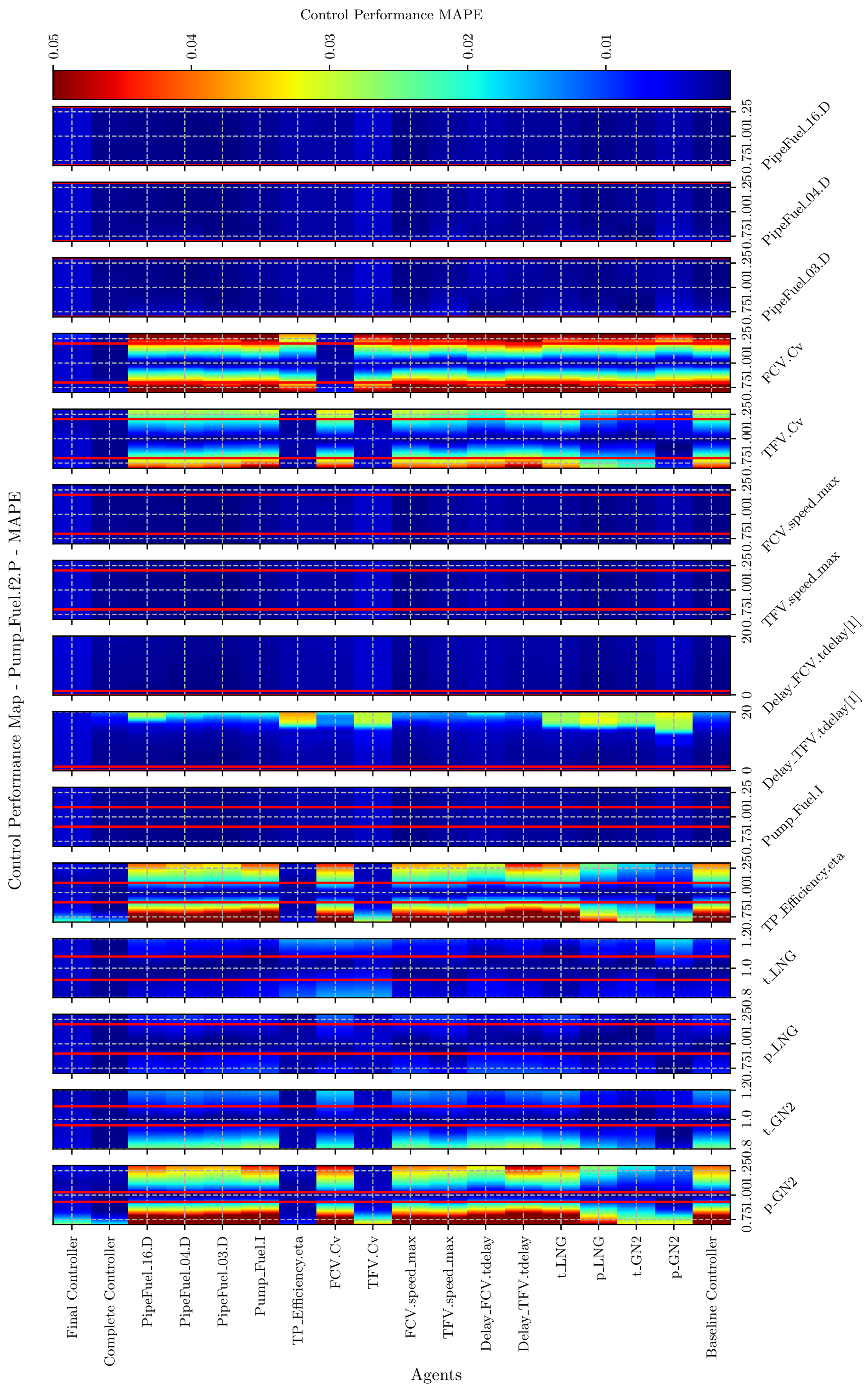


Figure D.1: DR Analysis for Pump Discharge Pressure - MAPE

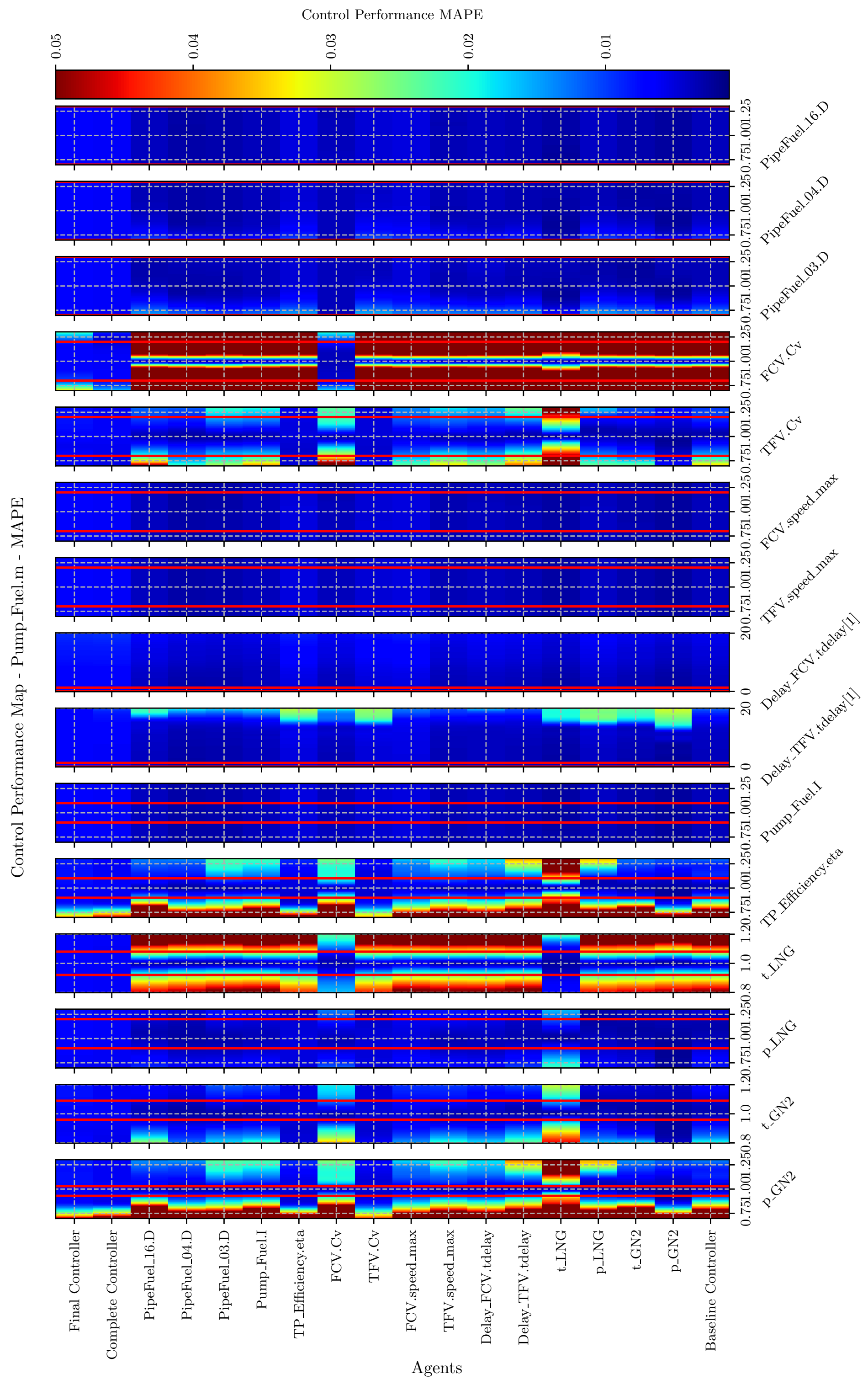


Figure D.2: DR Analysis for Pump Mass Flow - MAPE

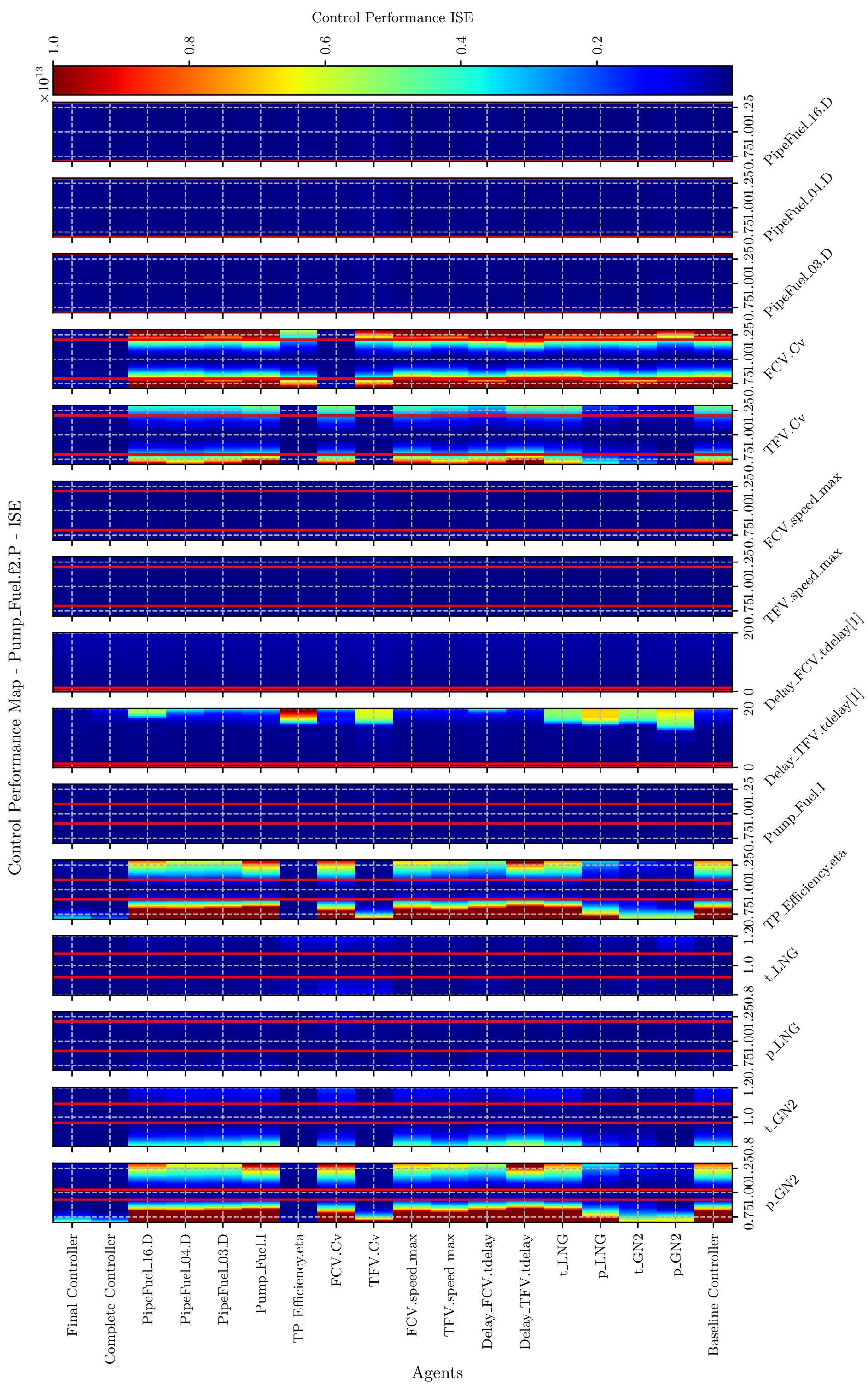


Figure D.3: DR Analysis for Pump Discharge Pressure - ISE

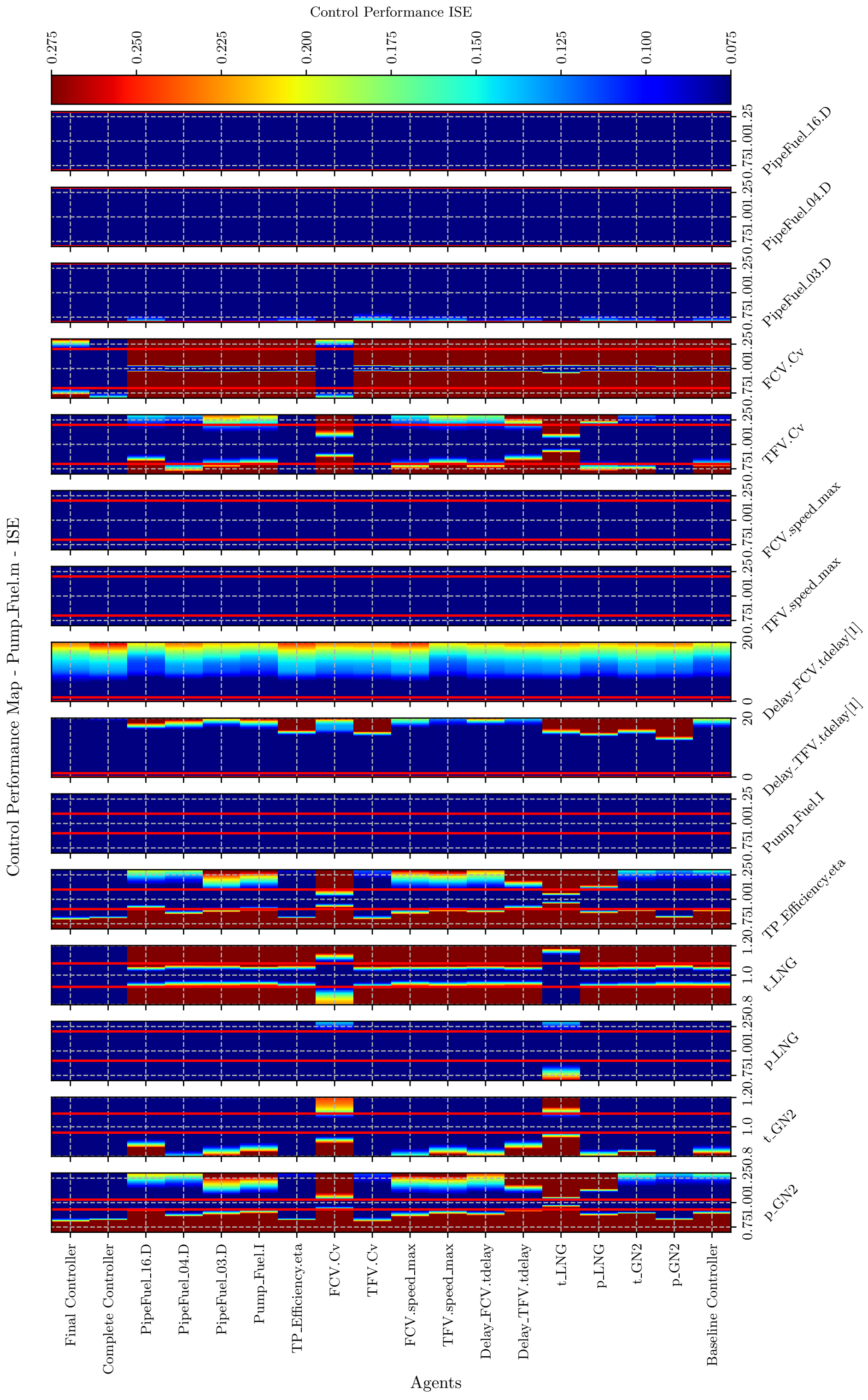


Figure D.4: DR Analysis for Pump Mass Flow - ISE

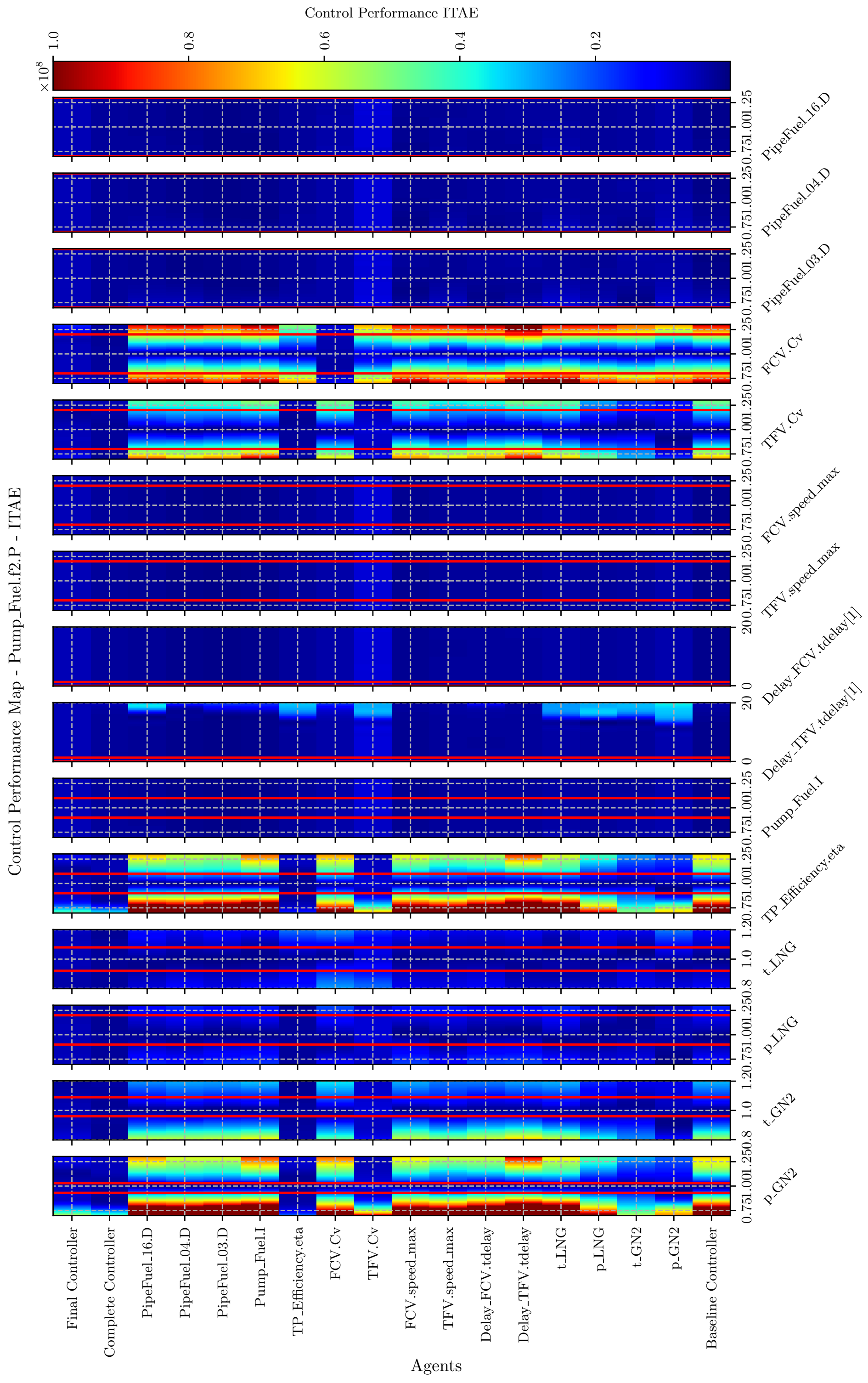


Figure D.5: DR Analysis for Pump Discharge Pressure - ITAE

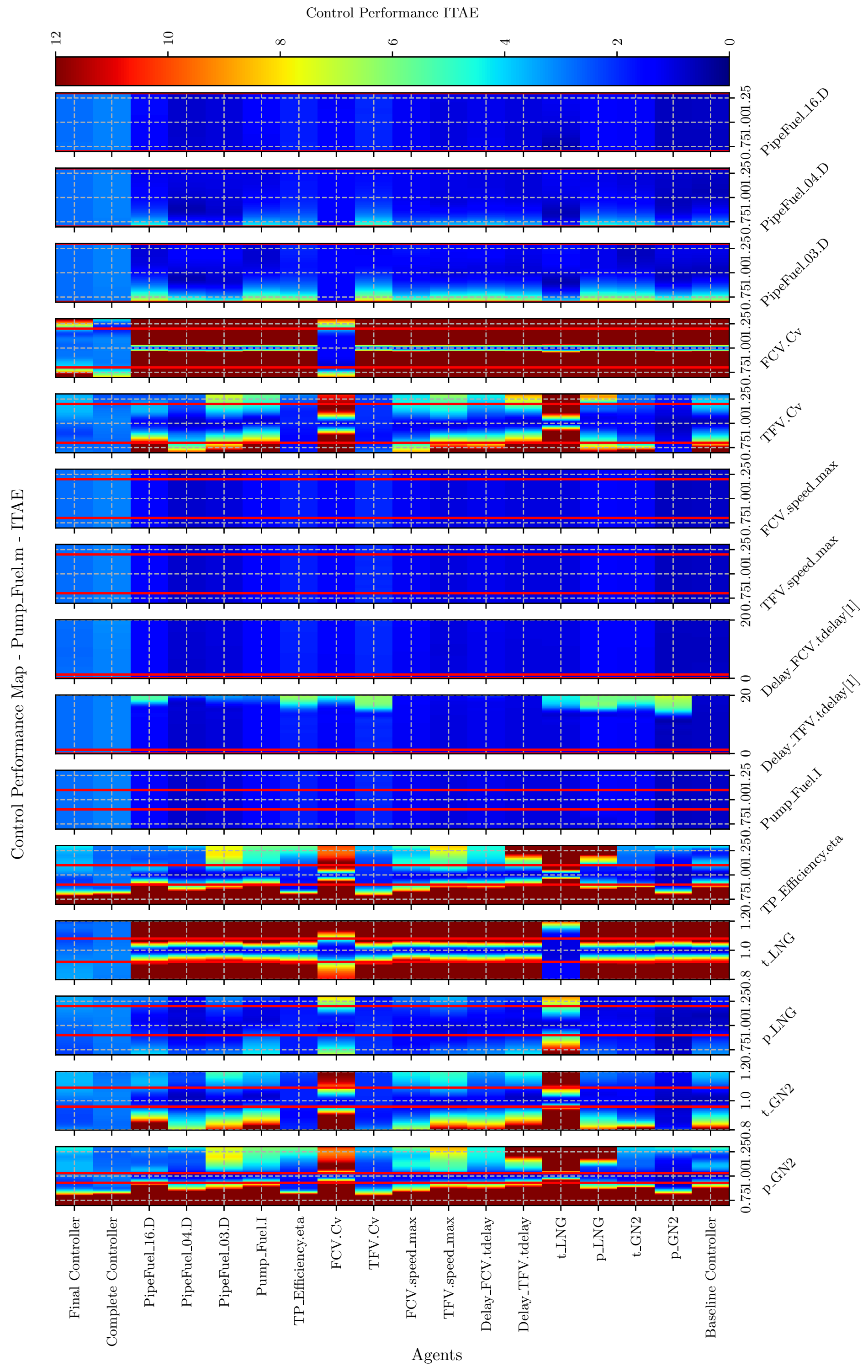


Figure D.6: DR Analysis for Pump Mass Flow - ITAE

Bibliography

[1] G.P. Sutton and O. Biblarz. *Rocket Propulsion Elements.* John Wiley & Sons, 2017.

[2] Dieter K. Huzel and David H. Huang. *Modern Engineering for Design of Liquid-Propellant Rocket Engines.* AIAA Progress in Astronautics and Aeronatics Volume 147, 1992.

[3] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MA: The MIT Press, Cambridge, 2 edition, 2018.

[4] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic Algorithms and Applications, 2018.

[5] Jan Lunze, editor. *Regelungstechnik 1.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[6] Kai Dresia, Günther Waxenegger-Wilfing, Robson Henrique Dos Santos Hahn, Jan C. Deeken, and Michael Oschwald. Nonlinear Control of an Expander-Bleed Rocket Engine using Reinforcement Learning. In *Space Propulsion 2020+1*, März 2021.

[7] Ray SAC Algorithm Documentation. `https://docs.ray.io/en/latest/rllib/rllib-algorithms.html#sac`. Accessed: 2022-08-22.

[8] J. Deeken, G. Waxenegger-Wilfing, and R. Dos Santos-Hahn. LUMEN Technical Specification (TS-DEMO).

[9] Tobias Traudt, Robson Henrique Dos Santos Hahn, Thomas Mason, Christian Mader, Jan C. Deeken, Michael Oschwald, and S. Schlechtriem. LUMEN Turbopump - Design and Manufacturing of the LUMEN LOX and LNG Turbopump components. In *32nd International Symposium on Space Technology and Science*, 2019.

[10] Carl F. Lorenzo, Walter C. Merrill, Jeffrey L. Musgrave, and Asok Ray. Controls concepts for next generation reusable rocket engines. *Proceedings of the American Control Conference*, 6:3942–3950, January 1995. Proceedings of the 1995 American Control Conference. Part 1 (of 6) ; Conference date: 21-06-1995 Through 23-06-1995.

[11] Sergio Pérez-Roca, Julien Marzat, Hélène Piet-Lahanier, Nicolas Langlois, François Farago, Marco Galeotta, and Serge Le Gonidec. A survey of automatic control methods for liquid-propellant rocket engines. *Progress in Aerospace Sciences*, 107:63–84, 2019.

[12] Günther Waxenegger-Wilfing, Kai Dresia, Jan C. Deeken, and Michael Oschwald. Machine Learning Methods for the Design and Operation of Liquid Rocket Engines - Research Activities at the DLR Institute of Space Propulsion. In *Space Propulsion 2020+1*, März 2021.

[13] J. Degrave, F. Felici, and J. et al. Buchli. Magnetic control of tokamak plasmas through deep reinforcement learning., 2022.

[14] Günther Waxenegger-Wilfing, Kai Dresia, Jan Deeken, and Michael Oschwald. A Reinforcement Learning Approach for Transient Control of Liquid Rocket Engines. *IEEE Transactions on Aerospace and Electronic Systems*, 57(5):2938–2952, oct 2021.

[15] Karina Einicke. Mixture Ratio and Combustion Chamber Pressure Control of an Expander-Bleed Rocket Engine with Reinforcement Learning, 2021.

[16] Till Hörger. Reinforcement Learning Framework zur optimalen Regelung von Orbitalantrieben unter Berücksichtigung von Robustheit und Betriebsbereichseinschränkungen. Master's thesis, Universität Stuttgart, 2021.

[17] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of Real-World Reinforcement Learning, 2019.

[18] Jan C. Deeken, Günther Waxenegger-Wilfing, Michael Oschwald, and S. Schlechtriem. LUMEN Demonstrator - Project Overview. In *Space Propulsion 2020+1*, März 2021.

[19] A. Iannetti, N. Girard, D. Tchou-kien, C. Bonhomme, N. Ravier, and E. Edeline. *PROMETHEUS, A LOX/LCH4 REUSABLE ROCKET ENGINE.* 2017.

[20] Stephen D. Heister, William E. Anderson, Timothé L. Pourpoint, and R. Joseph Cassady. *Rocket Propulsion.* Cambridge Aerospace Series. Cambridge University Press, 2019.

[21] G. Waxenegger-Wilfing, K. Dresia, J. C. Deeken, and M. Oschwald. Heat Transfer Prediction for Methane in Regenerative Cooling Channels with Neural Networks. *Journal of Thermophysics and Heat Transfer*, 34(2):347–357, 2020.

[22] Jan Haemisch, Dmitry Suslov, Günther Waxenegger-Wilfing, Kai Dresia, and Michael Oschwald. LUMEN - Design of the Regenerative Cooling System for an Expander Bleed Cycle Engine Using Methane. In *7th International Space Propulsion Conference*, 2021.

[23] Kai Dresia, Simon Jentzsch, Günther Waxenegger-Wilfing, Robson Dos Santos Hahn, Jan Deeken, Michael Oschwald, and Fabio Mota. Multidisciplinary Design Optimization of Reusable Launch Vehicles for Different Propellants and Objectives. *Journal of Spacecraft and Rockets*, 58(4):1017–1029, 2021.

[24] G. Fischer. *Turbopump Systems for Liquid Rocket Engines NASA SP-8107.* NASA, August 1974.

[25] Hideto Kawashima, Akihide Kurosu, Teiu Kobayashi, and Koichi Okita. *Progress of LE-9 Engine Development.* 2018.

[26] RocketLab Ltd. RocketLab Website - Completed Missions. `https://www.rocketlabusa.com/missions/completed-missions/`. Accessed: 2022-07-21.

[27] Robson Henrique Dos Santos Hahn, Jan C. Deeken, Tobias Traudt, Michael Oschwald, Stefan Schlechtriem, and Hideto Negishi. LUMEN Turbopump - Preliminary Design of Supersonic Turbine. In *32nd International Symposium on Space Technology and Science*, 2019.

[28] Jim Swetye. Pump Sizing 101 - Reading a Centrifugal Pump Curve. `https://www.pumpsandsystems.com/pumps/reading-centrifugal-pump-curve`. Accessed: 2022-08-20.

[29] P. F. Seitz and R. F. Searle. Space Shuttle Main Engine Control System. In *National Aerospace Engineering and Manufacturing Meeting*. SAE International, feb 1973.

[30] Till Hörger, Kai Dresia, Günther Waxenegger-Wilfing, Lukas Werling, and Stefan Schlechtriem. Development of a test infrastructure for a neural network controlled green propellant thruster. In *8th Space Propulsion Conference*, Mai 2022.

[31] Fabio Muratore, Fabio Ramos, Greg Turk, Wenhao Yu, Michael Gienger, and Jan Peters. Robot Learning from Randomized Simulations: A Review, 2021.

[32] Garud N. Iyengar. Robust Dynamic Programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[33] Janosch Moos, Kay Hansel, Hany Abdulsamad, Svenja Stark, Debora Clever, and Jan Peters. Robust Reinforcement Learning: A Review of Foundations and Recent Advances. *Machine Learning and Knowledge Extraction*, 4(1):276–315, 2022.

[34] Martin Riedmiller. 10 Steps and Some Tricks to Set up Neural Reinforcement Controllers. In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade*, volume 7700 of *Lecture Notes in Computer Science*, pages 735–757. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[35] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960.

[36] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2015.

[37] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, 2018.

[38] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, 2017.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing Atari with Deep Reinforcement Learning. *CoRR*, abs/1312.5602, 2013.

[40] G. E. Uhlenbeck and L. S. Ornstein. On the Theory of the Brownian Motion. *Phys. Rev.*, 36:823–841, Sep 1930.

[41] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods, 2018.

[42] Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning Invariant Feature Spaces to Transfer Skills with Reinforcement Learning, 2017.

[43] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe Exploration in Continuous Action Spaces, 2018.

[44] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey. 2020.

[45] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2018.

[46] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World, 2017.

[47] Daniel J. Mankowitz, Nir Levine, Rae Jeong, Yuanyuan Shi, Jackie Kay, Abbas Abdolmaleki, Jost Tobias Springenberg, Timothy Mann, Todd Hester, and Martin Riedmiller. Robust Reinforcement Learning for Continuous Control with Model Misspecification, 2019.

[48] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning Dexterous In-Hand Manipulation, 2018.

[49] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active Domain Randomization, 2019.

[50] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik's Cube with a Robot Hand, 2019.

[51] Suneel Belkhale, Rachel Li, Gregory Kahn, Rowan McAllister, Roberto Calandra, and Sergey Levine. Model-Based Meta-Reinforcement Learning for Flight With Suspended Payloads. *IEEE Robotics and Automation Letters*, 6(2):1471–1478, apr 2021.

[52] Arnab Nilim and Laurent El Ghaoui. Robust Control of Markov Decision Processes with Uncertain Transition Matrices. *Operations Research*, 53(5):780–798, 2005.

[53] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a Posteriori Policy Optimisation, 2018.

[54] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust Adversarial Reinforcement Learning, 2017.

[55] Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations. 2020.

[56] Katsuhiko Ogata. *Modern control engineering.* Prentice-Hall electrical engineering series. Instrumentation and controls series. Prentice-Hall, Boston, 5th ed. edition, 2010.

[57] Johanna Holsten. *Gütekriterien zur Bewertung der Eigenschaften von Tiltwingflugzeugen zur Auslegung ihrer Basisregelung.* Dissertation, RWTH Aachen University, Aachen, 2017. Veröffentlicht auf dem Publikationsserver der RWTH Aachen University; Dissertation, RWTH Aachen University, 2017.

[58] Jan Deeken and Günther Waxenegger-Wilfing. LUMEN: engine cycle analysis of an expander-bleed demonstrator engine for test bench operation. In *Deutscher Luft- und Raumfahrtkongress 2016*, 2016.

[59] J. Moral, R. Vara, J. Steelant, and M. Rosa. ESPSS Simulation Platform. 05 2010.

[60] J. Moral, J. Ruiz, M. Aranda, and F. Rodríguez. ESPSS 3.6.0 User Manual, 2022.

[61] Sergio Pérez-Roca, Julien Marzat, Hélène Piet-Lahanier, Nicolas Langlois, Marco Galeotta, and Franç Farago. Model-Based Robust Transient Control of Reusable Liquid-Propellant Rocket Engines. *IEEE Transactions on Aerospace and Electronic Systems*, 57(1):129–144, 2021.

[62] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.

[63] Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning, 2020.

[64] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A Distributed Framework for Emerging AI Applications, 2017.

[65] Eric Liang, Richard Liaw, Philipp Moritz, Robert Nishihara, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica. RLlib: Abstractions for Distributed Reinforcement Learning, 2017.

[66] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica. Tune: A Research Platform for Distributed Model Selection and Training, 2018.

Proclamation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

[illegible]

Würzburg, August 2022