DEVELOPMENT OF A TEST INFRASTRUCTURE FOR A NEURAL NETWORK CONTROLLED GREEN PROPELLANT THRUSTER

ESTORIL, PORTUGAL | 09 – 13 MAY 2022

Till Hörger [(1)], Kai Dresia [(2)], Günther Waxenegger-Wilfing [(3)], Lukas Werling[(4)], Stefan Schlechtriem[(5)]

[(1)] Institute of Space Propulsion – Satellite and Orbital Propulsion, German Aerospace Center (DLR), 74239 Hardthausen, Germany, Till.Hoerger@dlr.de
[(2)] Institute of Space Propulsion – Rocket Propulsion Systems, German Aerospace Center (DLR), 74239 Hardthausen, Germany, Kai.Dresia@dlr.de
[(3)] Institute of Space Propulsion – Rocket Propulsion Systems, German Aerospace Center (DLR), 74239 Hardthausen, Germany. Professor for digital methods in the modeling and control of space propulsion, Institute of Computer Science, University of Würzburg, 97074 Würzburg, Germany, Guenther.Waxenegger@dlr.de
[(4)] Institute of Space Propulsion – Satellite and Orbital Propulsion, German Aerospace Center (DLR), 74239 Hardthausen, Germany Lukas.Werling@dlr.de
[(5)] Institute of Space Propulsion – Head of Insitute, German Aerospace Center (DLR), 74239 Hardthausen, Germany, Stefan.Schlechtriem@dlr.de

KEYWORDS: reinforcement learning, machine learning, artificial intelligence, rocket engine control, green propulsion, neural networks, test bench, python, nitrous oxide, ethane, control, test facility

ABSTRACT:

This paper describes the setup of a test facility for various modern control methods for a green propellant thruster. The overarching goal is the intelligent control of a rocket engine, for example with a neural network. Therefore, a flexible and secure training and testing environment is needed that offers the possibility to test and compare different control approaches. In this paper, particular attention is paid to the interface between the test facility and the neural network. A reinforcement learning based controller is described in more detail and in the last section the results of a first test run with a neural network-based controller at the test facility M11.5 are discussed.

1. Motivation for Intelligently Controlled Rocket Engines

The development of artificial intelligence methods, especially machine learning algorithms, has shown huge success in the last years [1]. Consequently, machine learning methods are used in order to solve real word engineering problems. Autonomous robots [2, 3], autonomous driving [4], speech recognition, picture classification and the flight control of drones [5] are famous examples for the application of machine learning. Also, in the aerospace industry more applications seem conceivable [6]. One the one hand the methods can be used to accelerate the development process of spacecraft, on the other hand artificial intelligence methods offer the possibility to intelligently control space craft and rocket engines [7]. So far the German aerospace center (DLR) uses neural networks e.g. for the heat transfer prediction in rocket engine cooling channels [8] and fatigue life estimation [9]. Furthermore the automatic detection of suitable precursors of combustion instabilities [10], highly efficient data analytics and computably inexpensive design studies are subject of investigations [11]. In all this applications safety, repeatability, interpretability and certification of the machine learning method must be ensured, which is a current field of research.

In the last years several use cases for throttleable rocket engines became relevant. Obviously, the propulsive landing technology as demonstrated by SpaceX is one well-known use case [12]. Several projects in Europe are ongoing to develop vertical take-off and landing technologies [13, 14, 15]. Moreover, lander applications for either robotic or manned missions to the moon, mars and beyond show the need for throttleable thrusters and a suitable control system thereof. The soft landing on another celestial body requires a precise thrust regulation, deep throttling and restart capabilities. Simultaneously the reuse of rocket engines due to cost savings [16] demands optimally controlled transients for lifetime expansion and robust control to handle degradation of components over their lifetime. The benefits of an intelligent control system for optimal transients and condition monitoring have already been investigated in the space shuttle area [17, 18]. Since then the algorithms for machine learning and the computing capabilities made an enormous progress. While these studies were limited to simulations and the SSME (space shuttle main engine) was controlled by a conventional PID controller, today it is possible to apply machine learning and especially neural network-based controllers to real applications.

For rocket engine control several publicly known control methods are applicable. Starting from the well-known PI and PID control, moving to LQR (linear-quadratic regulator) and linear model predictive control (MPC) [19] all these methods require a linearized state space model [20, 21].

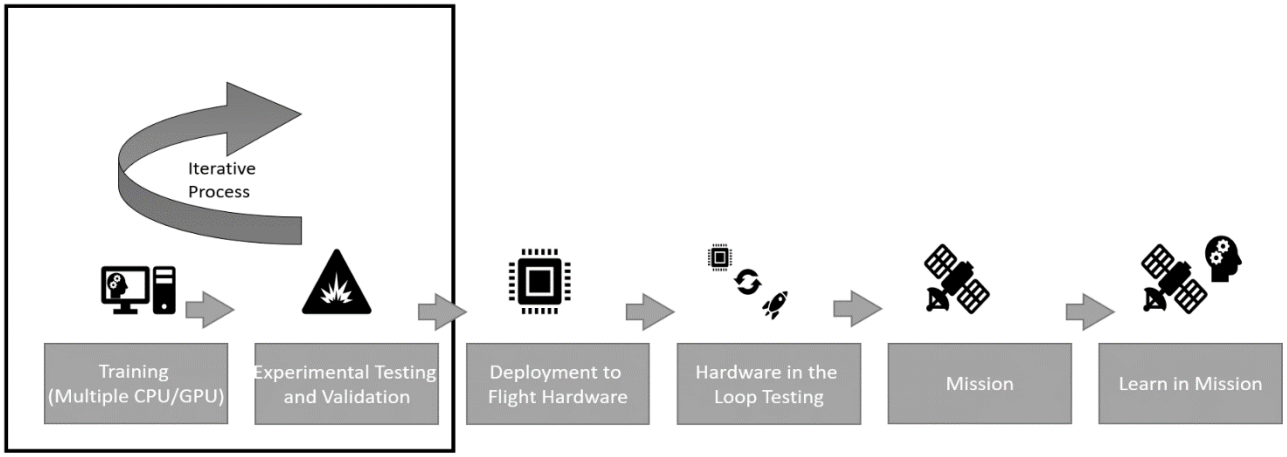

Figure 1 Development logic

Methods relying on machine learning and data driven methods like neural network based MPC or reinforcement learning (RL) trained controllers [22, 23] can be constructed directly from the nonlinear simulation model and therefore provide an optimal control for a wide range of system states.
Such advanced control methods allow to include an appropriate damage model in the controller or to keep wall temperatures at a given thrust under a certain level. It is possible to react to system changes due to reuse of the engine where the system behaviour changes due to component wear. And it's possible to develop controllers for optimal transients in start-up and shutdown of the engine [22].
One design approach to construct optimal closed - loop controllers investigated at DLR is the fast-growing field of deep reinforcement learning (DRL) [24], which is a subfield of machine learning. DRL promises many advantages compared to traditional control methods. Nonlinear simulations models can be used directly to derive the control law, no order reduction, linearization or derivation of a state space model is needed. After the training the response time of the controller is very short and the possibilities to include intelligence in the control law are practically unlimited. Nevertheless, one main drawback in DRL is, that until now the mathematical convergence of the algorithms is not fully understood [25, 26]. The outcome of the training process of an DRL algorithm is a neural network with tuned weights. The internal process of decision-making in artificial neural networks is not easy understandable by humans, therefore a DRL-based controller hast to face extensive testing before it comes into application. Fig. 1 shows a possible development process of a neural network-based controller for a spacecraft application. The first step is the training in a simulation environment. When successful results can be produced in simulation, extensive testing at real test facilities is needed. After that the trained neural network needs to be formatted and transferred to the flight hardware. Hardware in the loop tests will be conducted to ensure the functionality and safety [27]. After the deployment it is possible to further optimize the controller with data from the mission and establish learning and improvement during the whole life cycle of the product.
The next section will give a shot overview about the theoretical backgrounds of deep reinforcement learning. Following that section, the current test case and the test facility is described. Hereafter the connection of the test facility hardware and the neural network is highlighted. Finally results of a preliminary test with a closed loop control for constant combustion chamber pressure is presented. An outlook about the planned development of DRL controlled thrusters is given.

2. Reinforcement Learning

Reinforcement Learning is a subfield of machine learning. Generally, in machine learning a large amount of training data is used to generate a decision-making process. This field can be divided in three categories, depending on the quality of information give in the problem [28]. Supervised learning can be applied if the data set contains the input and the desired output data. The goal is to find the corresponding mapping rule between input and output, that can be used for example for image classification. Whereas in unsupervised learning the training data doesn't contain the target outputs. The goal here is to discover structures, similarities or hidden patterns in the input. Such algorithms can be used e.g. for cluster analysis [29].
Reinforcement learning (RL) is different from supervised und unsupervised learning. A so-called agent learns self-employed on the fly from simulation or real-world data. No predefined training dataset is needed. RL operates in discrete timesteps. The underlaying scheme of RL is visualized in Fig. 2. An agent interacts in every

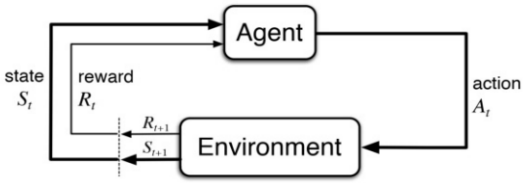

Figure 2 Reinforcement learning scheme [24]

timestep t with an environment with defined states s_t. In every timestep the agent gets information about the system state from the environment. The state is rated with a scalar value called reward r_t. The general idea is that the agent optimizes a decision rule, called policy, in a way to maximize the cumulative reward. In deep RL the policy is represented by a deep neural network that acts as function approximator. It maps a state of the system to an action a_t. The agent develops a strategy to maximize the cumulative reward $G(\tau) = \sum r_t$ that it gets. The reward is calculated by a user defined reward function that allows a rating of the state. The cumulative reward $G(\tau)$ is the sum of all single reward values in one episode, where the episode is one sequence of simulation. Through interaction with the environment the agent gets the information, encoded in the reward, which action is best in which situation. To avoid the cumulative reward for long episodes growing to infinity, a discount factor $0 \leq \gamma \leq 1$ is introduced (See Eq. 1).

$$G(\tau) = \sum_{t=0}^{\infty} \gamma^t r(s_t) \qquad \textit{Eq.1}$$

The value of γ defines how much future rewards come into account for the current action because γ is raised to the power of t. The process can be seen as a trial and error method with integrated feedback in the form of reward. Tab. 1 describes important expressions in RL.

Table 1 Key concepts in Reinforcement Learning

Expression	Description
Agent	Term for the algorithm or the controller. It optimizes the policy and computes the action in dependence of the state
Environment	System that interacts with the agent. In this case the rocket thruster
State	Physical state of the system
Reward	Scalar value that rates the state of the system. It is calculated by a user defined reward function
Action	Control output of the agent, it is sent to the environment
Observation space	All variables that serve as input values and system description for the agent.
Policy	Decision rule of the agent. It is a function that maps states to actions. It defines how the agent reacts to different system states. In deep RL the policy is an artificial neural network.

RL is a general framework to solve a Markov decision processes (MDP) [24]. A MDP is a model for a decision process that consist of a tuple (s, a, P, r, γ), where $P = P(s_t|s_{t+1}, a_{t+1})$ denotes the state transition function and describes the systems dynamics. The so-called Markov property says that every state transition in time is independent from the systems state before. A follow up of several MDP creates a trajectory τ or episode, that can be described as a sequence of state, action, reward $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots s_n, a_n, r_n)$

In the literature a wide variety of RL-algorithms with different learning paradigms, different advantages and disadvantages (deep Q-learning [30], policy gradient methods [31] or actor critic methods [32, 33]) can be found. The investigations at DLR are conducted with the soft actor critic (SAC) Algorithm [34]. Compared to on-policy algorithms like PPO [35] or A3C [36], SAC is characterized by a comparatively high sample efficiency which leads to faster training success. Furthermore, in comparison to off-policy algorithms like DDPG [32] or TD3 [33], the training process is stable and less hyperparameter tuning is needed. On-policy algorithms need completely new training samples after a policy update, while off-policy algorithms can learn from past samples using replay buffers.

In SAC beside the cumulative reward, also the entropy (as a measure of how random the agent acts) of the policy is maximized. In this way the task is completed successfully while acting as random as possible [34]. By that the policy is encouraged to exploration while being robust concerning model and estimation errors [34].

One drawback in the use of DRL is that, even with the use of a sample efficient algorithm like SAC, a huge amount of training data is needed to train the policy. This can be in the order of several million timesteps of training. To save cost and because of the limited availability of the test bench the training takes place in a simulation environment. Another reason why simulation-based training is beneficial is that the agent possibly enters dangerous system states during exploration and therefore may pose a threat for the test facility or test specimen. For the use-case of rocket engine control, EcosimPro ESPSS [37] is a suitable simulator. A challenge that comes with the training in simulation is to overcome the so-called sim to real gap [38]. Despite carful tuning of the simulation there will always be small deviations from the real system. The problem is that the RL algorithm learns to make use of these simulation inaccuracies and the application in the real world will fail. A method to overcome this is domain randomization [39, 40]. Here the simulation settings, like roughness, efficiency or pressure loss are changed during the training process and in this way the robustness of the policy concerning changes in the environment is increased, as the real environment becomes a subset of the randomized simulation [41].

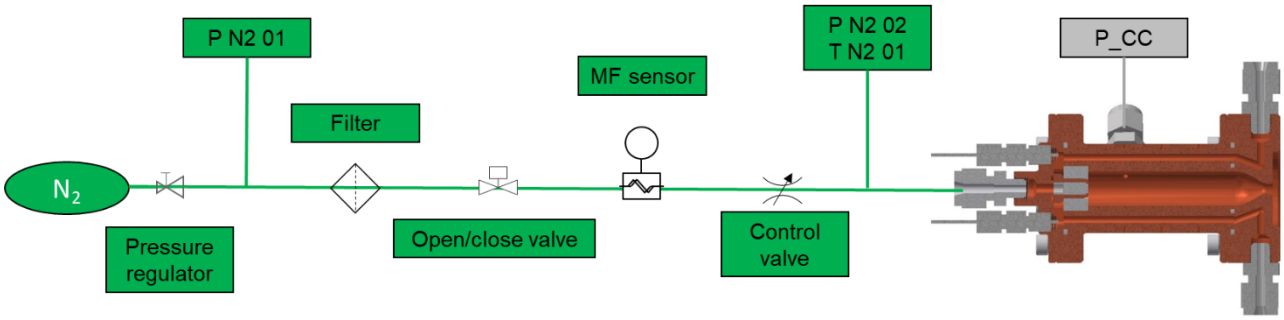

Figure 3 Piping and instrumentation diagram [23]

3. Testcase and Test Setup

The overall goal is to control a complete rocket propulsion system with RL. A suitable application is for example a $22\,N$ thruster fuelled with nitrous oxide and ethane. Such a system can be constructed either as premixed monopropellant- or traditional bipropellant system [42], while for the control task, the bipropellant system will be preferred as it offers more control options. Nitrous oxide/ethane offers a green alternative to the widely used satellite propellants hydrazine, monomethylhydrazine and dinitrogen tetroxide [43]. The latter will be attempted to replace due to their toxic, carcinogenic and environmentally harmful properties. At DLR several encouraging green propellants are under investigation [44, 45, 46, 47, 43]. Nitrous oxide/ethane are comparably cheap, widely accessible and offer a high $I_{sp} > 300\,s$. Because of the high vapour pressure of fuel and oxidizer it is possible to operate the system in self pressurized mode [43]. The in house designed combustion chamber [48] allows the adjustment of a wide range of mixture ratios and combustion chamber pressures with a throttle range from $35 - 130\%$ [47]. A control-system thereof promises a more efficient operation and allows, also with decreasing vapor pressure, a constant thrust operation of the combustor.

Since the RL-control method was never before demonstrated in the field of combustion chamber pressure control, a reduced system is designed for the first tests. Due to higher safety in the absence of combustion, a nitrogen cold gas thruster is chosen for preliminary investigations. The P&ID can be seen in Fig 3. Nitrogen is feed by a $200\,\mathrm{bar}$ central supply. The nitrogen pressure is reduced to a maximum of $100\,\mathrm{bar}$ by a PID controlled automatic pressure regulator of type Tescom ER5000. An open/close valve is used to start and stop the flow. After the valve a Coriolis mass flow meter is installed, followed by a proportional control valve of type m-tech MPG 03 PR. The nitrogen is then expanded through a chamber, initially designed for use with nitrous oxide and ethane.

The goal of the controller is to set the chamber pressure to the desired value by changing the position of the control valve.

All testing is conducted at DLR Lampoldshausen M11.5 test position [49, 50].

Three pressure sensors, P N2 01, P N2 02 and P_CC (see Fig. 3), serve as observation space for the controller. Furthermore, the current position of the control valve, the desired chamber pressure and the error as numeric difference between the desired pressure and the measured P_CC is fed as input to the neural network. Based on these values the network calculates an action value. This value is sent back to the test facility and changes the position of the control valve. The change in position is limited to 5% in each timestep, due to low movement speed of the valve. Table 2 gives an overview about the input and output data used by the controller.

Table 2 Input and output data used by the controller

Input value (observation space)	Output value (action space)
P N2 01 P N2 02 P CC Valve position Desired P CC Pressure-error	Adjustment of the valve position, clipped between -5 and $5\,\%$ per timestep

Fig. 4 visualizes the whole process of training, testing and application of an RL based controller. The neural network is trained in a feedback loop between the simulator and the reinforcement learning algorithm. In this case we use a carefully tuned simulation model of the cold gas-system described before. The simulation platform is Ecosimpro ESPSS [37]. An interface allows to exchange data between EcosimPro and python. In python the SAC implementation in the ray framework [51] is used to optimize the policy. When the training is completed a first test is conducted in the simulation environment (not visualized in Fig. 4). If this test is successful, the policy is exported to a pytorch neural network [52]. The network is then transferred to the test facility control computer. There it is included in the control program, which is also implemented in python. The analogue sensor

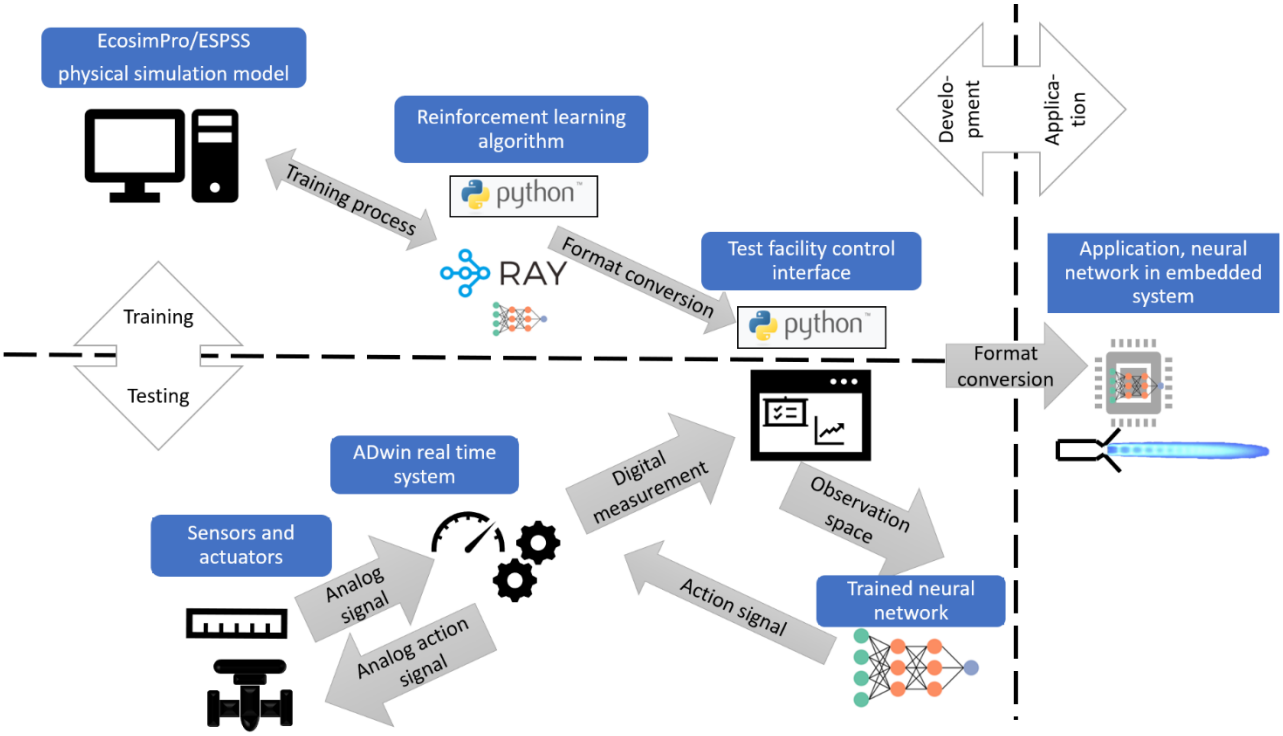

Figure 4 Training, testing and application process

signals, for example from the pressure transducers, are amplified and send to the real time capable ADwin system. Here the signal undergoes AD (analog-to-digital) conversion and is feed into a FIFO (first in, first out) memory. The ADwin system provides an interface to the python programming language, so it is possible to read all measurement values saved in the FIFO memory by the python test facility control program. In python data conversion, saving and plotting is organized. Derived values like the error of the chamber pressure are calculated. The observation space data is feed through the neural network in every timestep. In this way an output signal for the new commanded position of the control valve is generated. This signal is sent back from the python program to the ADwin system and further to the valve. If the testing and development is completed successfully, the format of the neural network can be converted once again. Then the network is in a compatible format for micro controller and could be implemented in a lightweight and space proven control system. This system can be qualified further for example in hardware in the loop tests [27].

4. The Control Interface of the Test Facility

The ADwinPro II system offers an interface to the python programming language. This way it is possible to control the test facility with a python code. Since “hard” real time programming with windows and especially python is difficult to achieve (i.e. due to python garbage collector or windows interrupts), the combination of the ADwin system, that provides real time measurement data, and python with its enormous amount of open source libraries for machine learning, is ideal.

In the control program a QTimer runs to timeout every $100\ \mathrm{ms}$. The timeout is connected to data acquisition, data processing, plotting and also feds the observation space data through the neural network for control purposes. Timing tests show, that for a control frequency of $10\ \mathrm{Hz}$ the timer has a deviation of $10\ \%$ in the mean. This is considered sufficient for the experiments with nitrogen. As shown in Tab. 3, a higher control frequency as for example $20\ \mathrm{Hz}$ cannot be accomplished because computation power for data acquisition, displaying and calculating the control output in parallel is too low, while a slower control loop is feasible.

Table 3 Performance of the system-timer

Control frequency	Mean [ms]	Min [ms]	Max [ms]
5 Hz	204,2	188,5	213,4
10 Hz	110,0	94,7	124,6
20 Hz	106,7	49,9	127,6

A $10\ \mathrm{Hz}$ control frequency appears to be low in a first view. Since the control valves used for the experiments, have a reaction time of about $300\ \mathrm{ms}$ and additionally an open time of about $2\ \mathrm{s}$ to fully open, $10\ \mathrm{Hz}$ are sufficient enough for this setup. In a later set up, faster valves should be used. The control frequency is then planned to be higher. For comparison, the engine controller of the space shuttle main engine operates at $50\ \mathrm{Hz}$ [53]. With faster valves included in this setup, it is planned to operate with a control frequency of $100\ \mathrm{Hz}$. Data acquisition and saving needs then to be put on a

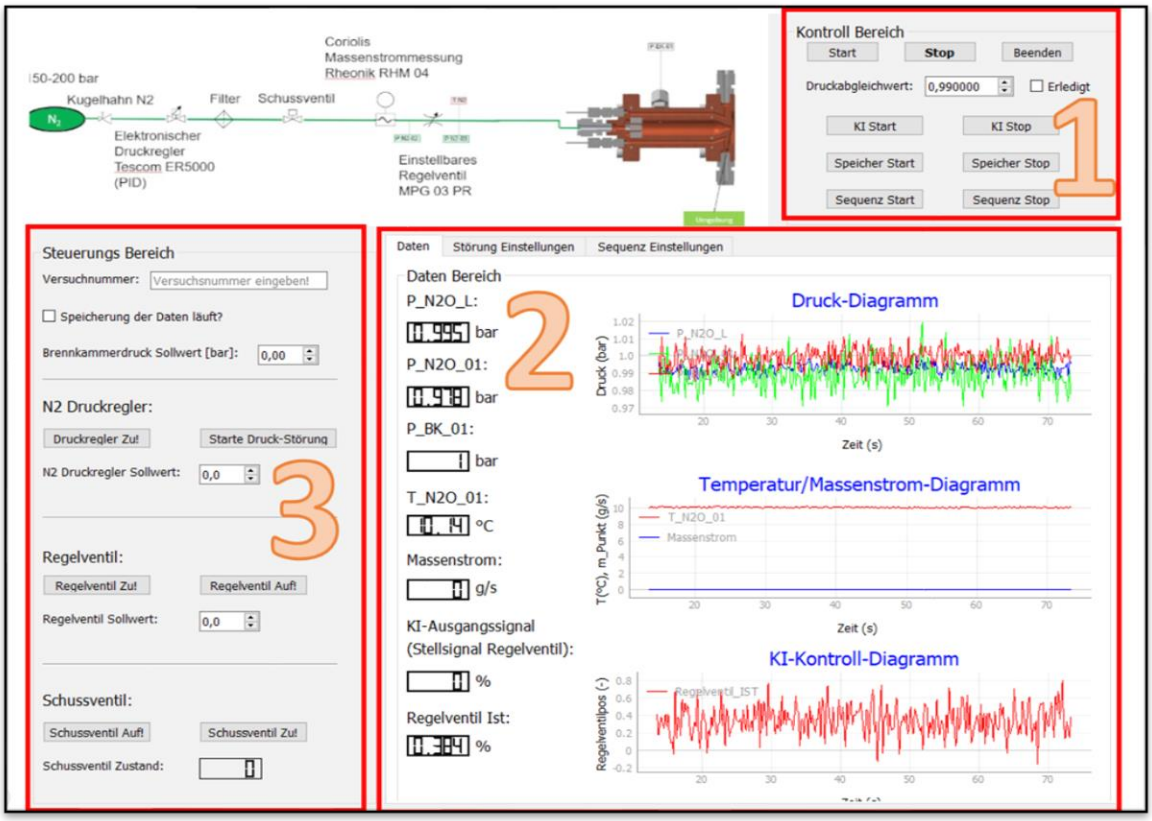

Figure 5 Screenshot from the test facility control program

second system, separate from the control system, to ensure a constant control frequency.

The advantage of using ADwin and python in parallel is that by using FIFOs in the ADwins system all measurement data is collected with the correct time stamp, even with a delayed timer in python, so no data is lost.

Python offers a number of advantages as a control system for test facilities: Firstly, no license costs arise. Secondly, there exists an extensive number of libraries to establish communication with all kind of sensors and other peripheral hardware with several bus systems. Thirdly the evaluation of the measurement data and plot creation can be integrated in the control program. No dedicated evaluation program is needed. The main reason for choosing python as framework for the control program was that there is no need for a further interface to include the neural network in the signal flow, as the neural network is represented originally in python.

Since the call of the neural network is a simple function call in python it is very easy to integrate and test different control methods at the test facility. For example, a model predictive controller and a PID controller could be implemented within minutes. This allows a fast comparison of the different control methods.

Fig. 5 shows a screenshot of the GUI (Graphical user interface) of the control program. The GUI is created with the open source Qt designer. The area marked with “1” in Fig. 5 allows to start and stop the whole program. Also, data acquisition and the start and termination of pre-programmed sequences can be controlled here. In a pre-programmed sequence pressure levels, data acquisition and the start and stop of the AI-control are started at defined times. In the sector labelled with “2” in Fig. 5, pressure, temperature, mass flow and valve position data are visualized with live-plots and LCD-displays. Also, the control output of the underlaying neural network, if activated, is represented. Section “3” in Fig. 5 allows to manually control all valves and pressure controllers in the set up to make system checks or conduct tests without an active sequence.

5. Constant Pressure Testcase

In the following section a first experiment with the neural network-based controller is presented. Fig. 6 shows a plot of the relevant data. The red line shows the desired chamber pressure. The first value to be achieved is $4\,\mathrm{bar}$. After 20 seconds of experiment time, the setpoint is changed to $7\,\mathrm{bar}$ and another 5 seconds later it is again changed to $5\,\mathrm{bar}$. 2.5 seconds after the experiment starts, the position of the control valve (solid blue line) is set to 10% open, while the open/close valve is still closed and the chamber pressure is at $1\,\mathrm{bar}$ ambient pressure. After 5.7 seconds the flow valve is opened (represented by the black vertical line) and nitrogen starts to flow while the controller is still inactive and the valve position is constant at about 10%. An influence of the flow on the position of the regulation valve can be seen, as it opens about 1% more after 6 seconds (highlighted with red circle in Fig. 6).

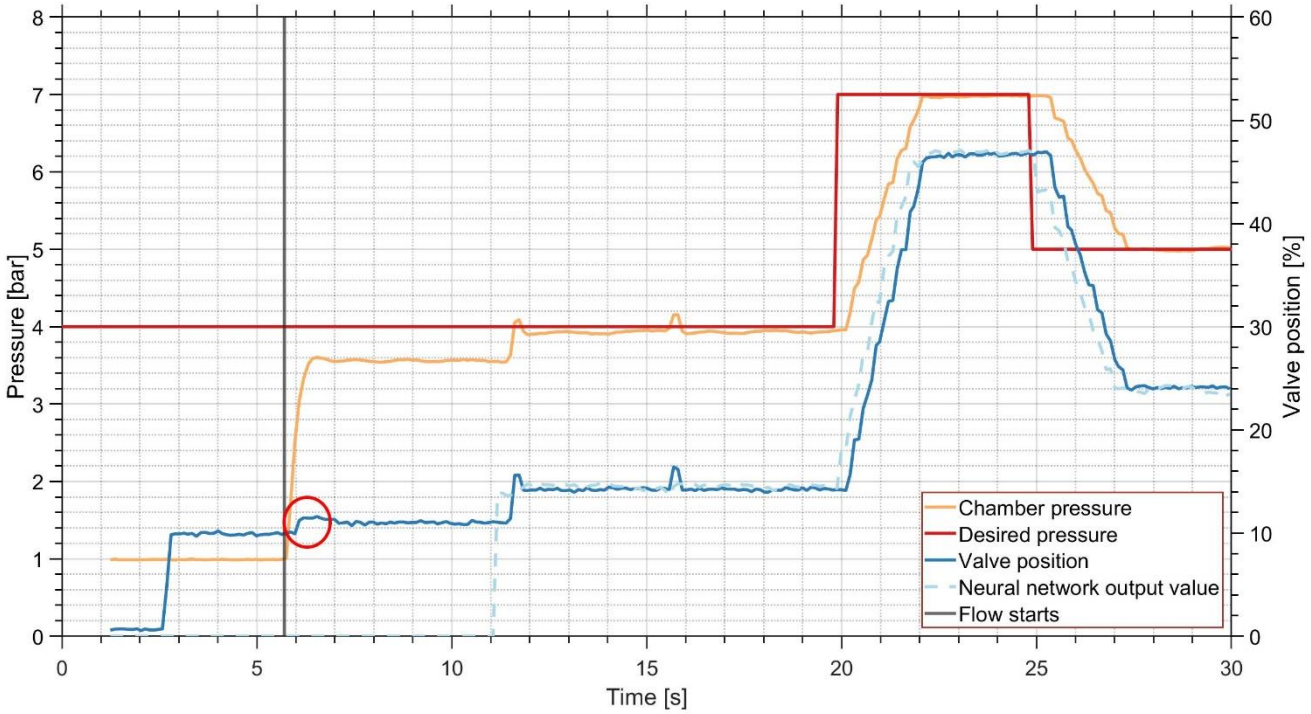

Figure 6 Experimental data

11.1 seconds after the experiment started, the controller is activated and starts changing the position of the control valve (dashed blue line). The time delay of the valve reaction to the setpoint can be observed, as the dotted line that represents the control signals runs ahead of the actual valve position. The neural network opens the regulation valve, while a deviation of mean $-0.06\ \mathrm{bar}$ to the setpoint remains.

Table 4 Statistical evaluation of the controller performance

Setpoint pressure [bar]	Settling time [s]	Mean squared error [bar]	Max deviation [bar]
4	0.44	0.0055	+0.15
7	2.08	0.0006	-0.04
5	2.30	0.0003	-0.03

The two sharp peaks in the valve position and corresponding in the chamber pressure at 12 and 16 seconds are possibly a result of valve hysteresis and are not reproducible. The step response from 5 to $7\ \mathrm{bar}$ is completed without overshoot or oscillations. The settle time to an error band of less than $0.1\ \mathrm{bar}$ appears to be slow, but with the limited valve speed, the restriction to a maximum change of 5% per timestep and the time delay in the valve response, a faster transition is physically not possible. Opposite to the $4\ \mathrm{bar}$ setpoint, $7\ \mathrm{bar}$ are reached nearly perfectly and no permanent deviation can be observed. The step from 7 to $5\ \mathrm{bar}$ is equivalent to the step response before. The time delay of the valve is once again visible as the deviation of the dashed and solid blue line.

6. Conclusion and Outlook, Towards an AI-controlled 200N Thruster

The example in the previous section shows that a neural network can be used as chamber pressure controller. Further experiments and comparison with other control methods like PID and model predictive control have to be conducted in order to evaluate the performance of the DRL control approach in competition with the others. Also experiments with external disturbances and changes in the system have to be conducted in order to evaluate the ability of the controller to react properly to changes in the system. This ability is referred as robustness and stability. The application of reinforcement learning and neural network-based software to safety critical systems - like rockets or spacecraft – has to meet high safety standards. A test concept or certification method for advanced control methods needs to be developed.

It is clear, that a simple nitrogen cold gas system can also be controlled with conventional methods, but this example was chosen as a first step to become familiar with the presented methodology. Reinforcement learning based controllers can show their advantages in more complicated systems. Therefore, it is planned to use the existing infrastructure and expand the controller to the $22\ \mathrm{N}$ nitrous oxide/ethane thruster. This step allows research at a more complex system. There will be two control valves (one for nitrous oxide and ethane respectively) and the control objective gets expanded to mixture ratio and temperature control. The controller will be used to enhance the efficiency of the thruster as it is always run at the optimal operating point. Through the inclusion of

combustion, the system gets harder to control, as combustion instabilities and a high dependency of the mixture ratio on system state make the dynamics more complex and faster.
New control valves need to be implemented in the test facility, as the current ones are slow, have a time delay and are too heavy for later flight application. In a further step, after mastering the $22\ \mathrm{N}$ system, it is planned to expand the control to a $200\ \mathrm{N}$ system. This will be much closer to a real application [54, 55], as a propulsion system in this thrust range can be used for a lander application on a robotic mission to mon or mars. The required higher mass flow rate for a upscaled system comes along with further challenges for the control system and it is anticipated that reinforcement learning based control can clearly demonstrate its advantages there.

List of Abbreviations

Acronym	Definition
A3C	Asynchronous advantage actor critic
AD	Analog digital converter
DDPG	Deep deterministic policy gradient
DLR	Deutsche Zentrum für Luft- und Raumfahrt e.V. (German aerospace center)
DRL	Deep reinforcement learning
ESPSS	European space propulsion system simulation
FIFO	First in first out
GUI	Graphical user interface
LQR	Linear-quadratic regulator
MDP	Markov decision process
MPC	Model predictive control
P&ID	Piping and instrumentation diagram
PI	Proportional-integral controller
PID	Proportional-integral-derivative controller
PPO	Proximal policy optimization
RL	Reinforcement learning
SAC	Soft actor critic
SSME	Space shuttle main engine

REFERENCES

[1] A. V. Joshi, Machine learning and artificial intelligence, Springer Verlag, 2020.

[2] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray and others, “Learning dexterous in-hand manipulation,” *The International Journal of Robotics Research,* vol. 39, p. 3–20, 2020.

[3] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba and L. Zhang, *Solving Rubik's Cube with a Robot Hand,* 2019.

[4] A. Folkers, Steuerung eines autonomen Fahrzeugs durch Deep Reinforcement Learning, Springer-Verlag GmbH, 2019.

[5] F. Sadeghi and S. Levine, *CAD2RL: Real Single-Image Flight without a Single Real Image,* 2017.

[6] G. Waxenegger-Wilfing, K. Dresia, J. Deeken and M. 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 Conference*, Virtual Event, 2021.

[7] G. Waxenegger-Wilfing, K. Dresia, J. Deeken and M. Oschwald, “A Reinforcement Learning Approach for Transient Control of Liquid Rocket Engines,” *IEEE Transactions on Aerospace and Electronic Systems,* no. 57, pp. 2938-2952, 2021, doi: 10.1109/TAES.2021.3074134.

[8] 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,* vol. 34, p. 347–357, 2020, doi: 10.2514/1.T5865

[9] K. Dresia, G. Waxenegger-Wilfing, J. Riccius, J. C. Deeken and M. Oschwald, “Numerically efficient fatigue life prediction of rocket combustion chambers using artificial neural networks,” in *Proceedings of the 8th European Conference for Aeronautics and Space Sciences*, 2019.

[10] G. Waxenegger-Wilfing, U. Sengupta, J. Martin, W. Armbruster, J. Hardi, M. Juniper and M. Oschwald, “Early detection of thermoacoustic instabilities in a cryogenic rocket thrust chamber using combustion noise features and machine learning,” *Chaos: An Interdisciplinary Journal of Nonlinear Science,* vol. 31, p. 063128, 2021.

[11] K. Dresia, S. Jentzsch, G. Waxenegger-Wilfing, R. D. Santos Hahn, J. Deeken, M. Oschwald and F. Mota, “Multidisciplinary design optimization of reusable launch vehicles for different propellants and objectives,” *Journal of Spacecraft and Rockets,* vol. 58, p. 1017–1029, 2021, doi: 10.2514/1.A34944.

[12] SpaceX, *Twitter, Stage one has landed,* 2015.

[13] I. Waugh, A. Davies, E. Moore and J. Macfarlane, “VTVL technology demonstrator for planetary landers,” 2016.

[14] E. Dumont, T. Ecker, C. Chavagnac, L. Witte, J. Windelberg, J. Klevanski and S. Giagkozoglou, “CALLISTO-Reusable VTVL launcher first stage demonstrator,” 2018.

[15] E. Dumont, S. Ishimoto, P. Tatiossian, J. Klevanski, B. Reimann, T. Ecker, L. Witte, J. Riehmer, M. Sagliano, S. Giagkozoglou, I. Petkov, W. Rotärmel, R. Schwarz, D. Seelbinder, M. Markgraf, J. Sommer, D. Pfau and H. Martens, “CALLISTO: a Demonstrator for Reusable Launcher Key Technologies,” in *32nd ISTS*, 2019.

[16] J. Foust, “SpaceX gaining substantial cost savings from reused Falcon 9,” 5 April 2017. [Online]. Available: https://spacenews.com/spacex-gaining-substantial-cost-savings-from-reused-falcon-9/. [Accessed 11 April 2022].

[17] W. Merrill and C. Lorenzo, “A reusable rocket engine intelligent control,” in *24th Joint Propulsion Conference*, 1988.

[18] C. F. Lorenzo, A. Ray and M. S. Holmes, “Nonlinear control of a reusable rocket engine for life extension,” *Journal of Propulsion and Power,* vol. 17, p. 998–1004, 2001.

[19] S. Pérez-Roca, J. Marzat, É. Flayac, H. Piet-Lahanier, N. Langlois, F. Farago, M. Galeotta and S. Le Gonidec, “An MPC approach to transient control of liquid-propellant rocket engines,” *IFAC-PapersOnLine,* vol. 52, p. 268–273, 2019.

[20] S. Pérez-Roca, J. Marzat, H. Piet-Lahanier, N. Langlois, M. Galeotta, F. Farago and S. Le Gonidec, “Model-based robust transient control of reusable liquid-propellant rocket engines,” *IEEE Transactions on Aerospace and Electronic Systems,* vol. 57, p. 129–144, 2020.

[21] S. Pérez-Roca, J. Marzat, H. Piet-Lahanier, N. Langlois, F. Farago, M. Galeotta and S. Le Gonidec, “A survey of automatic control methods for liquid-propellant rocket engines,” *Progress in Aerospace Sciences,* vol. 107, p. 63–84, 2019.

[22] G. Waxenegger-Wilfing, K. Dresia, J. C. Deeken and M. Oschwald, *A Reinforcement Learning Approach for Transient Control of Liquid Rocket Engines, IEEE Transactions on Aerospace and Electronic Systems,* no. 57, pp. 2938-2952, 2021, doi: 10.1109/TAES.2021.3074134.

[23] T. Hörger, K. Dresia, G. Waxenegger-Wilfing, L. K. Werling and S. Schlechtriem, “Preliminary Investigation of Robust Reinforcement Learning for Control of an Existing Green Propellant Thruster,” in *AIAA Propulsion and Energy 2021 Forum*, 2021.

[24] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction, second edition ed., vol. 2018: 1, Cambridge, Mass.: The MIT Press, 2018.

[25] J. Achiam, E. Knight and P. Abbeel, *Towards Characterizing Divergence in Deep Q-Learning,* arXiv, 2019.

[26] J. Fu, A. Kumar, M. Soh and S. Levine, “Diagnosing Bottlenecks in Deep Q-learning Algorithms,” in *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[27] G. Waxenegger-Wilfing, K. Dresia, M. Oschwald and K. Schilling, “Hardware-In-The-Loop Tests of Complex Control Software for Rocket Propulsion Systems,” in *71st International Astronautical Congress (IAC)*, 2020.

[28] G. Rebala, A. Ravi and S. Churiwala, An introduction to machine learning.

[29] A. Rüttgers, A. Petrarolo and M. Kobald, “Clustering of paraffin-based hybrid rocket fuels combustion data,” *Experiments in Fluids,* vol. 61, p. 1–17, 2020.

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, *Playing Atari with Deep Reinforcement Learning,* 2013.

[31] R. S. Sutton, D. McAllester, S. Singh and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” *Advances in neural information processing systems,* vol. 12, 1999.

[32] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, *Continuous control with deep reinforcement learning,* 2019.

[33] S. Fujimoto, H. van Hoof and D. Meger, *Addressing Function Approximation Error in Actor-Critic Methods,* 2018.

[34] T. Haarnoja, A. Zhou, P. Abbeel and S. Levine, *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,* 2018.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, *Proximal Policy Optimization Algorithms,* arXiv, 2017.

[36] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver and K. Kavukcuoglu, *Asynchronous Methods for Deep Reinforcement Learning,* 2016.

[37] E. A. Internacional, *EcosimPro 6.2.0,* 2020.

[38] W. Zhao, J. P. Queralta and T. Westerlund, *Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey,* 2020.

[39] B. Mehta, M. Diaz, F. Golemo, C. J. Pal and L. Paull, *Active Domain Randomization,* 2019.

[40] L. Weng, “Domain Randomization for

Sim2Real Transfer," *lilianweng.github.io/lil-log,* 2019.

[41] T. Dai, K. Arulkumaran, T. Gerbert, S. Tukra, F. Behbahani and A. A. Bharath, *Analysing Deep Reinforcement Learning Agents Trained with Domain Randomisation,* 2020.

[42] L. Werling, "Entwicklung und Erprobung von Flammensperren für einen vorgemischten, grünen Raketentreibstoff aus Lachgas und Ethen".

[43] L. Werling, T. Hörger, K. Manassis, D. Grimmeisen, M. Wilhelm, C. Erdmann, H. Ciezki, S. Schlechtriem, S. Richter, T. Methling, E. Goos, C. Janzer, C. Naumann and U. Riedel, "Nitrous Oxide Fuels Blends: Research on premixed Monopropellants at the German Aerospace Center (DLR) since 2014," in *AIAA Propulsion and Energy Forum 24.-26.08.2020*, 2020.

[44] M. Negri, M. Wilhelm, C. Hendrich, N. Wingborg, L. Gediminas, L. Adelöw, C. Maleix, P. Chabernaud, R. Brahmi, R. Beauchet, Y. Batonneau, C. Kappenstein, R.-J. Koopmans, S. Schuh, T. Bartok, C. Scharlemann, U. Gotzig and M. Schwentenwein, "New technologies for ammonium dinitramide based monopropellant thrusters – The project RHEFORM," *Acta Astronautica,* vol. 143, p. 105–117, 2018.

[45] M. Kurilov, C. Kirchberger, D. Freudenmann, A. D. Stiefel and H. Ciezki, "A Method for Screening and Identification of Green Hypergolic Bipropellants," *International Journal of Energetic Materials and Chemical Propulsion,* vol. 17, 2018.

[46] F. Lauck, J. Balkenhohl, M. Negri, D. Freudenmann and S. Schlechtriem, "Green bipropellant development–A study on the hypergolicity of imidazole thiocyanate ionic liquids with hydrogen peroxide in an automated drop test setup," *Combustion and Flame,* vol. 226, p. 87–97, 2021.

[47] L. Werling and T. Hörger, "Experimental analysis of the heat fluxes during combustion of a N2O/C2H4 premixed green propellant in a research rocket combustor," *Acta Astronautica,* vol. 189, p. 437–451, 2021.

[48] J. Dobusch, *Entwicklung und Tests von 3D-gedruckten, regenerativ gekühlten Versuchsbrennkammern für Raketentreibstoffe aus N2O und C2H6,* 2021.

[49] H. Ciezki, L. Werling, M. Negri, F. Strauss, M. Kobald, C. Kirchberger, D. Freudenmann, C. Hendrich, M. Wilhelm, A. Petrarolo and S. Schlechtriem, "50 Years of Test Complex M11 in Lampoldshausen – Research on Space Propulsion Systems for Tomorrow," 2017.

[50] M. Wilhelm, L. Werling, F. Strauss, F. Lauck, C. Kirchberger, H. Ciezki and S. Schlechtriem, "Test Complex M11: Research on Future Orbital Propulsion Systems and SCRamjet Engines," in *International Astronautical Congress*, 2019.

[51] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan and I. Stoica, *Ray: A Distributed Framework for Emerging AI Applications,* 2018.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox and R. Garnett, Eds., Curran Associates, Inc., 2019, p. 8024–8035.

[53] C. F. Lorenzo and J. L. Musgrave, "Overview of rocket engine control," in *AIP Conference Proceedings*, 1992.

[54] I. Waugh, E. Moore and J. Macfarlane, "Closed-loop throttle control of a N2O/IPA thruster," 2018.

[55] J. Wink, T. Knop, S. Powell and R. Werner, "DEVELOPMENT AND GROUND TESTING OF A 200 N VACUUM THRUST CLASS THRUSTER USING A NOVEL NITROUS OXIDE/PROPENE PROPELLANT COMBINATION," 2018.