Neural networks for minimum time satellite reorientation

Acta Astronautica 249 (2026) 445–462

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Research paper

Neural networks for minimum time satellite reorientation ☆

Luca Bigelli [a,*], Federica Paganelli Azza [b], Luca Romanelli [b], Mattia Varile [b], Marcello Romano [c]

[a] Department of Mechanical and Aerospace Engineering, Politecnico di Torino, Corso Duca degli Abruzzi 24, Turin, 10129, Italy
[b] AIKO S.r.l., Corso Stati Uniti 45, Turin, 10129, Italy
[c] Chair of Astrodynamics, Technical University of Munich (TUM), Willy-Messerschmitt-Str. 11, Taufkirchen, 82024, Germany

ARTICLE INFO

Keywords:
Satellite reorientation
Time-optimal slewing
Machine learning
Neural networks
Imitation learning

ABSTRACT

This work investigates the use of Guidance and Control Networks (G&CNETs) for time-optimal reorientation of an asymmetric rigid spacecraft. Classical optimal-control approaches generally require the online solution of computationally demanding optimisation problems, which may limit their applicability on resource-constrained platforms. Motivated by the promising performance of G&CNETs in related guidance and control applications, this study assesses their ability to approximate optimal slewing policies. To this end, several lightweight G&CNET architectures are designed and trained offline using expert trajectories generated by the GPOPS-II optimal control solver. Both regression- and classification-based formulations are considered, together with different activation functions. The resulting controllers are evaluated through an extensive simulation campaign in terms of manoeuvre time degradation relative to the optimal reference solution, robustness to spacecraft inertia uncertainties and navigation errors, and computational efficiency on commercial off-the-shelf embedded CPU platforms. Results show that the proposed networks achieve near-optimal manoeuvre times, with increases of only a few percent relative to the optimal reference solutions, representing a competitive alternative to state-of-the-art methods. Furthermore, the measured inference times of only a few milliseconds make them compatible with onboard execution on computationally limited hardware. These findings demonstrate the practical viability of G&CNET-based controllers for time-optimal spacecraft reorientation and highlight their potential for future autonomous spacecraft missions.

1. Introduction

1.1. Time-optimal spacecraft reorientation

Time-optimal satellite reorientation is crucial for many space missions. Historically, eigenaxis rotation was the aerospace standard as it represents the shortest angular path [1–6]. However, due to the nonlinear nature of Euler's dynamics, the minimum-angle path is generally not time-optimal. This was proven by Bilimoria and Wie [7], who demonstrated that the optimal manoeuvre for an inertially symmetric spacecraft with independently constrained torques exhibits a bang–bang structure with significant nutation. Furthermore, Fleming et al. [8] showed that constraining a satellite to its eigenaxis wastes nearly 20% of its manoeuvring capability. Subsequently, Bai and Junkins [9] clarified the theoretical boundaries of this problem, proving that eigenaxis manoeuvres are time-optimal exclusively for inertially symmetric spacecraft under spherical control constraints (bounded total magnitude). Conversely, the presence of independent actuator limits (cubic constraints) or asymmetric mass distributions inherently dictates non-eigenaxis optimal solutions.

1.2. Numerical solution of the optimal control problem

With the sole exception of the inertially symmetric spacecraft under magnitude-constrained torques [9], the time-optimal reorientation admits no closed-form analytical solution, and progressively more realistic configurations have therefore been addressed with numerical solvers. Shen and Tsiotras [10] solved the time-optimal reorientation of an axisymmetric spacecraft with two independent torques using a cascaded direct-indirect scheme. Subsequently, the fully asymmetric case was first solved by Proulx and Ross [11] employing the Legendre pseudospectral (PS) method within DIDO [12–14]. A feedback formulation was obtained by Ross et al. [15] through Carathéodory-π trajectories, in which the open-loop solution is continuously recomputed at high frequency; applied to time-optimal reorientations, this sample-and-hold

☆ This article is part of a Special issue entitled: 'SPAICE2025_invited only' published in Acta Astronautica.

* Corresponding author.
E-mail address: luca.bigelli@polito.it (L. Bigelli).

https://doi.org/10.1016/j.actaastro.2026.08.033
Received 31 March 2026; Received in revised form 30 July 2026; Accepted 5 August 2026
Available online 19 August 2026

scheme reached update rates of 5 Hz [8,16]. Although highly accurate, pseudospectral methods require substantial computational time to converge [17,18], and reducing the number of nodes to accelerate the process often yields suboptimal or infeasible solutions [19,20]. These computationally intensive algorithms are therefore unsuitable for resource-constrained onboard platforms. Consequently, current engineering practice dictates solving the optimal control problem offline on the ground and subsequently uploading the precomputed control sequence to the spacecraft [21,22].

1.3. Hybrid and sub-optimal approaches

To reduce the time required to solve the optimal control problem, hybrid two-stage schemes have been proposed, in which a heuristic optimiser, such as Particle Swarm or Bacterial Foraging Optimisation [23,24], provides a rough estimate that a pseudospectral solver then refines, cutting the overall computation time [18]. Heuristic optimisers have also been used as stand-alone solvers [25–27]. Both strategies have been employed mainly for reorientations subject to path constraints, such as the keep-out cones protecting sensitive payloads [28,29].

To mitigate onboard computational limitations, several sub-optimal approaches have also been proposed. Homotopic methods [30,31] continuously deform an easy-to-handle auxiliary problem, such as an energy-optimal manoeuvre, into the more complex time-optimal one. While guaranteeing high precision, they still require several seconds to compute the open-loop trajectory. Inverse Dynamics in the Virtual Domain (IDVD) [32–35] instead approximates the trajectory with analytical functions over a virtual domain, drastically reducing the number of optimisation variables and rapidly generating feasible solutions, at the price of smooth control profiles and therefore of manoeuvre times longer than the bang–bang optimum. Both families, however, still rely on the online solution of an optimisation problem, which remains challenging for limited onboard hardware.

1.4. Learning-based guidance and control

To bypass the computational burden of solving an optimisation problem, Machine Learning (ML) techniques have gained increasing attention. As documented in recent surveys [36–39], AI-based methods offer a paradigm shift for spacecraft Guidance, Navigation, and Control (GNC) systems by moving the effort of optimal control entirely to an offline training phase, so that online execution reduces to a fast neural network inference. Within this paradigm, several approaches exist that differ mainly in what the network is asked to represent and in how it is trained.

Supervised learning of the optimal state-feedback. Rooted in the early work on Behavioural Cloning (BC) [40–43], this family trains a network on state-control pairs extracted from optimal trajectories, so that guidance and control are solved simultaneously and the network maps states directly to raw optimal actions. Popularised in astrodynamics under the name of Guidance and Control Networks (G&CNETs) [44], the approach was first applied end-to-end to planetary landing by Sánchez-Sánchez and Izzo [45], and later extended to low-thrust interplanetary transfers [46] and time-optimal constant-acceleration rendezvous [47]. A parallel line of research has also addressed the problem of real-time optimal control, training networks on solutions of the indirect method for minimum-time solar-sail transfers [48], time-optimal asteroid landing [49], and fuel-optimal rocket powered landing [50]. Closer to the present application, Biggs and Fournier [51] trained a multilayer perceptron offline to reproduce a predictive controller minimising the total impulse, and applied it to the detumbling and the large-angle slew of a CubeSat. The same principle was applied by Chai et al. [52] to the atmospheric reentry of a hypersonic vehicle, with the objective of minimising the accumulated aerodynamic heating. It was also developed for quadrotors [53–55], demonstrating direct onboard execution without a traditional inner-loop tracking controller. More recently, periodic activation functions (SIRENs) [56] have been introduced in this context and shown to train faster and to reach a lower training error [57,58]. A related line of work uses the network as a component within an otherwise conventional architecture rather than as the controller itself, as in Tang et al. [59], where the predicted trajectory is refined online by a Quadratic Programming solver and tracked by a PID controller; in such schemes the optimality of the result remains mediated by the layer that consumes the network output. A known limitation of the whole family is that the learned policy can only be as good as the expert dataset used for training, so that particular care must be devoted to the dataset generation [60,61]. Moreover, at execution time the states visited by the system are those induced by the learned controller, which may be under-represented in the dataset and on which the network is consequently less accurate [62,63].

Deep Reinforcement Learning. Instead of imitating an expert, this family learns a policy through interaction with a simulated environment, guided by a scalar reward. In the context of satellite slew manoeuvres, Elkins et al. [64] developed an adaptive RL agent robust to unmodelled dynamics, parameter uncertainties and external disturbances. In a subsequent work the reward function was refined to achieve very high pointing accuracy [65]. Reinforcement learning has been applied to attitude control also by Vedant et al. [66], who trained a policy under randomised inertia, so that the resulting agent is capable of driving satellites with very different inertia tensors, and benchmarked it against a classical quaternion-rate-feedback controller. Beyond attitude, reinforcement meta-learning has been shown to adapt online to unmodelled environmental forces in landing and close-proximity scenarios [67,68], while Hovell and Ulrich [69] learn a guidance strategy that issues velocity commands to a conventional controller. In the quadrotor domain, end-to-end reinforcement learning has been applied to time-optimal flight, where a policy issuing direct motor commands was found to outperform one commanding thrust and body rates to a classical inner-loop controller [70]. In all these works, however, optimality is not imposed directly but induced through the reward [71]. Moreover, the problems addressed differ from the time-optimal reorientation of a rigid body: the reported metrics are pointing accuracy and robustness and, although near time-optimality is claimed and supported on isolated cases, it is not systematically assessed against the corresponding optimal manoeuvre times, as this was not the focus of those works.

The relative merits of BC and RL have recently been assessed directly. Federici et al. [60] compared them on a linear multi-impulsive rendezvous problem and found both viable, with BC requiring a sufficiently heterogeneous expert dataset to generalise and RL avoiding that requirement at a substantially higher training cost. More recently, Holt et al. [61] carried out a systematic comparison for continuous-thrust G&CNETs, concluding that BC yields more optimal solutions around nominal conditions whereas RL performs better out of distribution. The same authors observe that the results favouring RL obtained in the drone-racing domain do not transfer directly to spacecraft, where the dynamics are well modelled, disturbances are comparatively small, and optimality is paramount.

Approximate dynamic programming and adaptive neural control. In the works considered here the network is not trained offline but adjusted online, while the spacecraft operates. Leeghim et al. [72] add a neural term to a quaternion feedback law in order to estimate and cancel the effect of uncertain inertia and actuator misalignments. Dong et al. [73] and Yang et al. [74] instead approximate the solution of the Hamilton–Jacobi–Bellman equation from data collected online, encoding attitude-forbidden zones, angular-velocity limits and actuator misalignment directly into the cost. Stability is established by Lyapunov arguments, so that convergence of the reorientation error is guaranteed and the constraints are satisfied throughout the learning process. This, however, requires the adaptation law to be executed onboard alongside the control, rather than a single network inference. Moreover, the cost is a smooth function trading control effort against pointing error, so

that the resulting policy is optimal with respect to that cost rather than to the manoeuvre time.

Learning from the optimality conditions. A final family embeds the optimality conditions themselves into the learning problem, and its members differ in what the network is asked to represent. Nakamura-Zimmerer et al. [75] learn the value function of the Hamilton–Jacobi–Bellman equation, that is the optimal cost-to-go from any state, from which the feedback control is recovered by minimising the Hamiltonian, and demonstrate the approach on the attitude control of a rigid-body satellite over a fixed time horizon. D'Ambrosio and Furfaro [76] learn instead the state and costate histories by means of Pontryagin Neural Networks, trained so that the equations of the Minimum Principle are satisfied, and apply them to fuel-optimal interplanetary transfers and Mars landing; there, the discontinuous throttle is recovered by replacing the sign function with a hyperbolic tangent and applying a continuation procedure on the smoothing parameter. Related studies propose instead to learn the switching function associated with the bang–bang optimal control law rather than directly approximating the control signal itself [77], which may alleviate some of the difficulties associated with learning discontinuous control actions. In the works considered here, however, the discontinuity of the optimal control is handled by smoothing or regularising it, rather than by representing it directly.

1.5. Contributions

In the state of the art, Behavioural Cloning has been applied to reproduce the optimal feedback of several different systems, from optimal landings and transfers to agile quadrotor flight [45,46,57,58]. In several of these problems the underlying optimal control is bang–bang, and how its discontinuity is encoded at the network output is known to be a delicate design choice: continuous outputs have been reported to struggle in reproducing a bang–bang thrust, and discrete encodings to introduce uncontrolled transitions between the levels [49].

However, in none of these problems is the target an attitude: what is driven to a prescribed condition is the position of the vehicle, so that what is rendered time-optimal is a displacement rather than a reorientation. The time-optimal reorientation of a spacecraft, in which the target is an attitude and the motion is governed by Euler's equations of a fully asymmetric rigid body, remains unaddressed to the best of the authors' knowledge.

Conversely, learning-based methods have indeed been applied to spacecraft attitude control, but never under a minimum-time objective: the costs adopted are the total impulse [51], smooth functions trading control effort against pointing error [73–75], or a reward through which time-optimality can at most be induced [64,65]. Moreover, where near time-optimality is claimed, the degradation with respect to the exact optimal manoeuvre time is not quantified over a representative set of manoeuvres, so the actual cost of replacing an optimal controller with a learned one is not documented for this problem. The end-to-end Behavioural Cloning of a time-optimal reorientation policy, with its distance from optimality systematically quantified, appears therefore to be still lacking, and the present work aims to fill this gap.

Following the G&CNET approach, several lightweight architectures capable of accurately reproducing bang–bang control policies are designed and evaluated, with the manoeuvre time as the primary figure of merit: each controller is measured by the margin that separates it from the optimal duration, while retaining the ability to generate its command online. To achieve this, the general-purpose optimal control solver GPOPS-II [78] is employed as the expert demonstrator to generate exact time-optimal trajectories, which are subsequently used to extract labelled state–action pairs for offline neural network training.

With a specific focus on manoeuvre optimality, closed-loop reliability, and onboard feasibility, the contributions of the present study are fivefold:

1. An end-to-end Behavioural Cloning formulation of the time-optimal reorientation problem for a fully asymmetric spacecraft is proposed. Both regression- and classification-based output heads are considered, together with different activation functions, and their performance and generalisation capabilities are assessed over an extensive simulation campaign against the optimal solutions generated by GPOPS-II.
2. The proposed controllers are benchmarked against two state-of-the-art alternatives, a Reinforcement Learning agent and a Nonlinear Model Predictive Control scheme, both tuned to be as competitive as possible so as to ensure a fair comparison.
3. The robustness of all controllers is characterised beyond nominal conditions, under inertia uncertainty, additive sensor noise, and state-estimation errors.
4. The closed-loop behaviour is verified empirically by means of a Lyapunov-based analysis, and the failures observed in the test campaign are analysed in detail in order to identify the mechanism that causes them.
5. With a view towards deployment on resource-constrained platforms, the proposed architectures are validated on commercial off-the-shelf CPU-only embedded devices, assessing their compatibility with real-time onboard execution.

The rest of the paper is organised as follows. Section 2 defines the time-optimal reorientation problem and describes the dataset generation process. Section 3 introduces the neural architectures and the training procedure. Section 4 presents the test campaign: the nominal performance and the onboard inference time, the robustness to a tighter pointing requirement, to inertia mismatches and to navigation errors, and the comparison against the reinforcement learning and NMPC baselines. Section 5 then assesses the closed-loop stability of the trained controllers and identifies the mechanism behind the failures observed with the sinusoidal activation. Finally, Section 6 concludes the work by summarising the main findings.

2. Problem formulation

2.1. Dynamical system

We investigate rest-to-rest spacecraft reorientation manoeuvres [7,9], where the satellite starts and ends with zero angular velocity. The attitude dynamics are governed by Euler's rotational equations of motion:

$$\mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}) = \mathbf{T}, \tag{1}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity, $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the inertia matrix about the principal axes, $\mathbf{T} \in \mathbb{R}^3$ is the control torque vector, and $\times$ denotes the cross product. As a case study, we selected a 6U CubeSat platform that has no symmetry. It has dimensions $x = 0.20\,\mathrm{m}$, $y = 0.10\,\mathrm{m}$, $z = 0.30\,\mathrm{m}$ and mass 12.0 kg. The mass is uniformly distributed, so geometric axes align with the principal axes. The resulting diagonal inertia matrix is $\mathbf{J} = \mathrm{diag}(0.10,\ 0.13,\ 0.05)\,\mathrm{kg\ m^2}$. Maximum torque capability is 3.2 mNm along each body axis. As attitude kinematic parametrisation, quaternions are adopted to represent the spacecraft attitude in simulation. The attitude kinematics is described by the following differential equation [79]:

$$\dot{\mathfrak{q}} = \frac{1}{2}\mathbf{Q}(\mathfrak{q})\boldsymbol{\omega}, \tag{2}$$

where $\mathfrak{q} = [q_0\ q_1\ q_2\ q_3]^\top$ is the unit quaternion, with q_0 denoting the scalar component and $\mathbf{Q}(\mathfrak{q})$ is defined as

$$\mathbf{Q}(\mathfrak{q}) = \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}. \tag{3}$$

2.2. Optimisation problem

The time optimal rest-to-rest reorientation problem can be stated as: find the optimal control input that steers the satellite's attitude from an initial orientation $\boldsymbol{\sigma}_0$ to a desired final orientation $\boldsymbol{\sigma}_f$, while minimising the duration of the manoeuvre [7]. The associated cost function is:

$$J = \int_{t_0}^{t_f} dt = t_f, \tag{4}$$

subject to the spacecraft's rotational dynamics and kinematics described in Eqs. (1) and (2) and to the actuator torque constraints:

$$\mathrm{T}_{\min,i} \leq \mathrm{T}_i(t) \leq \mathrm{T}_{\max,i}, \quad i = x, y, z, \tag{5}$$

indicating that each component of the control torque vector is bounded independently. These constraints are commonly referred to as *cubic constraints*, as they define a cubic admissible region in $\mathbb{R}^3$ for the control torque vector [7,9].

2.3. Dataset generation

A total of 100,000 rest-to-rest reorientation trajectories are generated by numerically solving the previously presented time-optimal control problem using GPOPS-II. These trajectories form the basis of the dataset employed for the training of the proposed neural network architectures. Each manoeuvre begins from a rest condition, with the satellite's body frame aligned with the Earth-Centred Inertial (ECI) frame. The final orientation is defined by a rotation axis uniformly sampled over the unit sphere and a rotation angle drawn uniformly from the interval $[-\pi, \pi]$. These boundary conditions are used in GPOPS-II to compute the corresponding time-optimal state and control profiles. Since GPOPS-II outputs are non-uniformly sampled in time, all trajectories are resampled at uniform 0.05 s intervals using Piecewise Cubic Hermite Interpolation (PCHIP). To validate the interpolation fidelity, each control profile is propagated forward in time. Control trajectories yielding a final attitude error greater than 1.5 deg with respect to the original target attitude are discarded. Furthermore, in view of the bang–bang nature of the time-optimal solution, trajectories containing one or more control samples in the interval $[-0.3 \cdot \mathrm{T}_{\max},\ 0.3 \cdot \mathrm{T}_{\max}]$ are excluded, in order to retain only manoeuvres with predominantly fully saturated control profiles. Although the theoretical optimal control is bang–bang, some trajectories generated by GPOPS-II were observed to contain a limited number of non-fully-saturated samples and the interpolation step further accentuated this effect. For this reason, the value $0.3 \cdot \mathrm{T}_{\max}$ was selected empirically after qualitatively considering different thresholds, as it provided a reasonable trade-off between preserving the bang–bang character of the dataset and avoiding the rejection of an excessive number of trajectories due to a small number of non-fully-saturated samples. Although the initial axis-sampling guarantees uniform distribution of target orientations, the outlined operations introduced a bias. A balancing step was thus applied to restore uniform coverage of the unit sphere while minimising the number of discarded trajectories.

The constructed dataset included 41,751 optimal trajectories, yielding a total of 8,235,939 samples. Each sample consists of a state–action pair $[\mathbf{x}_{\mathrm{err}}\ \mathbf{u}_{\mathrm{opt}}]^\top$, where the spacecraft error state is defined as $\mathbf{x}_{\mathrm{err}} = [\mathrm{q}_{\mathrm{err}}\ \boldsymbol{\omega}_{\mathrm{err}}]^\top$. The attitude quaternion error is computed as $\mathrm{q}_{\mathrm{err}} = \mathrm{q}^{-1} \otimes \mathrm{q}_{\mathrm{ref}}$, where q^{-1} is the inverse of the current attitude quaternion, $\mathrm{q}_{\mathrm{ref}}$ is the target quaternion, and $\otimes$ denotes quaternion multiplication. The angular velocity error is defined as $\boldsymbol{\omega}_{\mathrm{err}} = \boldsymbol{\omega}_{\mathrm{ref}} - \boldsymbol{\omega}$, where $\boldsymbol{\omega}$ is the current angular velocity and $\boldsymbol{\omega}_{\mathrm{ref}} = \mathbf{0}$ is the reference angular velocity, which is the zero vector since rest-to-rest manoeuvres are considered. The corresponding action is given by the optimal control torque vector $\mathbf{T}_{\mathrm{opt}} = [\mathrm{T}_x\ \mathrm{T}_y\ \mathrm{T}_z]^\top$. This dataset forms the basis for training the proposed neural network architectures.

3. Neural network design

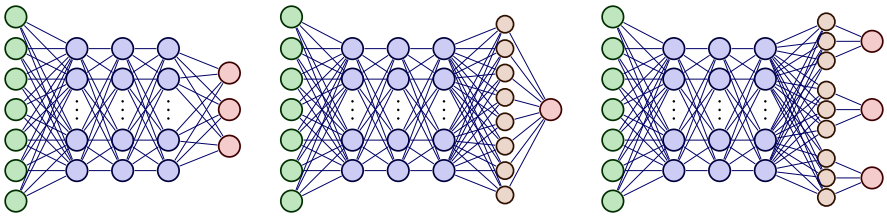
3.1. Architectures

Several neural network architectures have been designed, trained, and tested. Each network receives as input the spacecraft state error, as previously defined. Quaternions are preferred as attitude inputs because their components are naturally bounded in $[-1, 1]$, which is beneficial for network training. All proposed models share a common backbone designed to remain simple, lightweight, and strictly grounded in established literature. Based on the extensive hyperparameter analysis performed in [53] and the successful flight performances demonstrated in [54], three hidden layers with 100 units each were chosen. Previous studies confirmed that this specific network size provides an excellent trade-off between state-feedback approximation accuracy and network complexity. Having fixed the network capacity, our analysis primarily focuses on the output layer configuration and the activation functions. Specifically, we evaluated architectures using Rectified Linear Unit (ReLU), Softplus, and sinusoidal activation functions [58]. Architectures employing sinusoidal activations are commonly referred to as SInusoidal REpresentation Networks (SIREN), as introduced in [56]. Inspired by previous works [53,54], the first architecture proposed is a regression-based model (Fig. 1(a)) with three sigmoid outputs, each mapped to the range $[-\mathrm{T}_{\max}, \mathrm{T}_{\max}]$ to produce continuous control values. However, standard regressors are known to struggle with discontinuous bang–bang control profiles. To overcome this limitation and fully exploit the inherently discrete nature of optimal bang–bang profiles, we reformulate the control approximation as a classification task. Accordingly, the second proposed architecture (Fig. 1(a)) formulates the control prediction problem as three independent binary classification tasks, one for each control axis. This architecture adapts the baseline regressor by modifying its output mapping and replacing the loss function accordingly. The third architecture (Fig. 1(b)), which will be referred to as "8-Class", frames the control prediction as an 8-class classification problem. Each class represents a unique combination of the three control components, where each component can take on values of either $-\mathrm{T}_{\max}$ or $+\mathrm{T}_{\max}$. A softmax output layer is employed to select the most probable control combination among the eight classes. Lastly, the fourth architecture (Fig. 1(c)) predicts each control component independently using a three-class classifier, assigning one of the values in $\{-\mathrm{T}_{\max},\ 0,\ +\mathrm{T}_{\max}\}$. Although this formulation does not strictly align with the theoretical bang–bang control profile, as it includes the 0 control action, it demonstrated strong performance in validation and testing, supporting its inclusion in the study. This architecture will be referred to as "ClassMultiOutput" (ClMO) in the following.

3.2. Training

The proposed architectures were trained using the ADAM optimiser [80]. A learning rate of 10^{-4} was adopted for networks with ReLU and Softplus activations, while a reduced value of 10^{-5} was used for the sinusoidal activation function. All models were trained with a batch size of 512 samples. The training process was initially set to 500 epochs, but extended to 1000 epochs after observing that the loss had not yet plateaued and that the additional epochs led to improved performance across all configurations. No learning rate scheduler was employed. Regarding the loss and evaluation metrics, Mean Squared Error (MSE) was used as the loss function for the regression architecture, with Mean Absolute Error (MAE) employed as the performance metric. For the classification models, accuracy was used as the metric. Sparse categorical cross-entropy and binary cross-entropy were used as evaluation metrics for the multi-class classifiers and the binary classifier, respectively.

Training and validation histories of the loss functions and the corresponding metrics are shown in Fig. 2. In terms of training behaviour,



(a) Regressor and Binary Classifier Architecture.

(b) Multi-Class (8-Class) Classifier Architecture.

(c) Multi-Output Multi-Class Classifier Architecture (ClassMultiOutput).

Fig. 1. Proposed neural network architectures. Green, blue, and red units represent input, hidden, and output layer units, respectively. (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

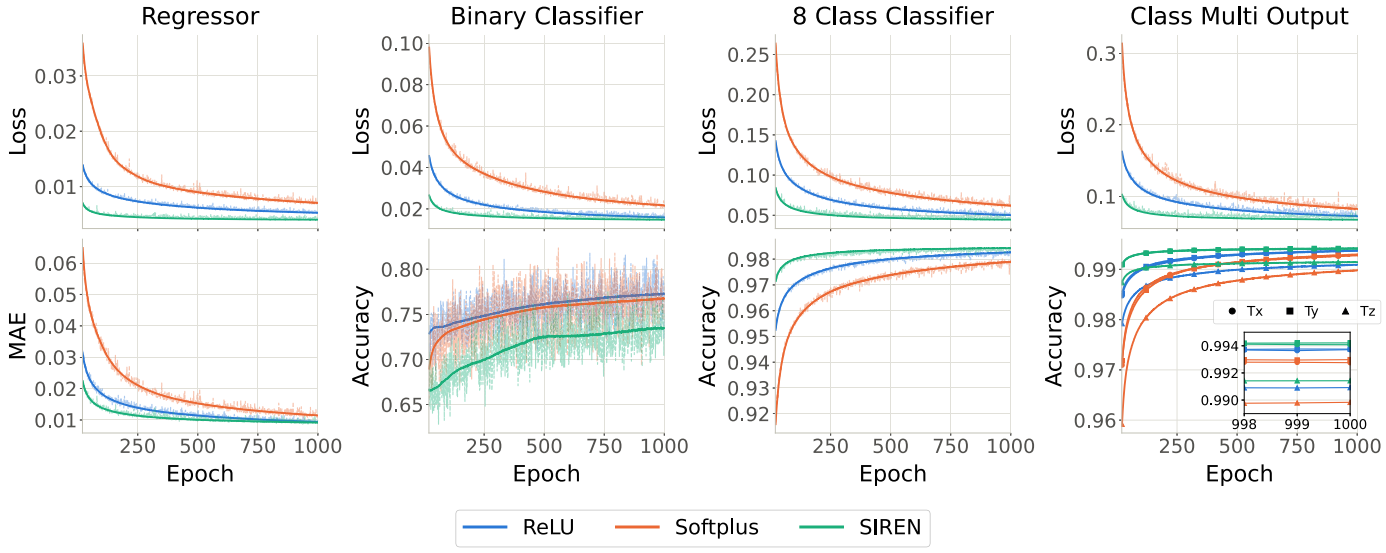


Fig. 2. Training and validation loss and performance metrics for the four G&CNET architectures (columns). Top panels show the loss histories, while bottom panels report the corresponding metrics. For the Multi-Output Multi-Class classifier, separate metrics are shown for the three control components, indicated by different markers: circle for T_x, square for T_y, and triangle for T_z. To improve readability, validation accuracy is omitted in this case, whereas for the other architectures it is shown as a dashed line. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

SIREN networks achieved the lowest training and validation losses, slightly outperforming those with ReLU, whereas Softplus consistently resulted in higher loss and validation values. For the 8-Class classifier (Fig. 2, third column), accuracy exceeded 98% with ReLU and sinusoidal activations, and remained slightly below 98% with Softplus. The Class Multi Output classifier (Fig. 2, rightmost column) achieved above 99% accuracy across all three control components for all the activation functions, with one exception: the T_z output using Softplus, which was slightly below 99%. It is worth noting that the T_z control component consistently achieved marginally lower accuracy compared to T_x and T_y across all activation functions. As for the Binary Classifier, the validation accuracy exhibited a particularly noisy trend during training, and the final training accuracy remained lower compared to the other classifiers. Despite lower loss values, the sinusoidal activation yielded in this case the lowest accuracy (73%), compared to ReLU (77%) and Softplus (76%).

4. Tests and results

4.1. Test settings

The trained neural networks are evaluated within a closed-loop spacecraft attitude control system, as depicted in Fig. 3. In this configuration, each network acts as a controller, receiving the state error as input and producing the corresponding control torque vector as output. Unless stated differently, the state is assumed to be fully observable and no external disturbances are considered. In particular, orbital disturbances are neglected [61], as the manoeuvre duration is sufficiently short that their effect on the dynamics is negligible. The control loop is executed with a fixed time step of 0.05 s, consistent with the sampling interval used to interpolate the trajectories generated by GPOPS-II.

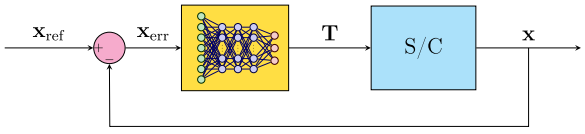


Fig. 3. Closed-loop spacecraft attitude control system.

To assess the generalisation capability of the proposed models, a total of 18,432 rest-to-rest reorientation manoeuvres are simulated. The manoeuvres are selected to ensure a uniform distribution of rotation axes over the unit sphere, while rotation angles are uniformly sampled in the interval $[-180, 180]$ deg, excluding the range $[-10, 10]$ deg. This threshold is introduced to exclude small-angle manoeuvres, for which linearised attitude dynamics provide sufficient accuracy and well-established linear controllers can be employed instead [81]. The

total number of tested manoeuvres arises from the adopted sampling strategy. To obtain a uniform distribution of rotation axes, the unit sphere is discretised using HEALPix (Hierarchical Equal Area isoLatitude Pixelization) [82], which partitions the spherical surface into equal-area regions. In particular, the sphere is subdivided into 3072 regions, and six manoeuvres are sampled from each region while satisfying the previously defined angular constraints, leading to a total of 18,432 test manoeuvres.

A manoeuvre is deemed successful (and thus terminated) if the final attitude error is below an angular accuracy threshold of 0.5 deg (0.00873 rad) and the final angular velocity magnitude is below 0.5 deg/s (0.00873 rad/s). Conversely, any manoeuvre whose duration is at least twice the optimal time computed by GPOPS-II, i.e. a percentage increase higher than 100%, is classified as a failure. This relative criterion is adopted for the G&CNETs and the RL agent. Preliminary tests, however, revealed that the NMPC exhibits substantially larger manoeuvre-time increases relative to GPOPS-II; applying the same 100% relative threshold would therefore flag an excessive fraction of manoeuvres as failures even under nominal conditions. For this reason, an NMPC manoeuvre is instead classified as a failure only when its duration exceeds an absolute value of 60 s, a limit that under nominal conditions still allows the vast majority of trajectories to converge. Although these accuracy levels may seem large, the goal of the proposed neural networks is to complete the slewing manoeuvre, i.e. a large-angle reorientation, up to that accuracy. As is typical in space applications, once a predefined pointing accuracy is achieved, the slew manoeuvre is considered complete, and the system transitions from the large-angle attitude manoeuvre (slew) to a fine-pointing or attitude-hold mode [81].

For comparison, besides the time-optimal baseline GPOPS-II, two additional methods are considered, both able to produce close-to-optimal solutions at a reduced computational cost and therefore suitable for online implementation: a Reinforcement Learning (RL) agent, specifically trained to solve the time-optimal reorientation problem, and a Nonlinear Model Predictive Controller (NMPC). The reader is referred to Appendices A and B for details on their implementation.

Since the aim of this work is to present controllers that generate quasi-optimal solutions while being lightweight enough to run onboard platforms with limited computational resources, two performance metrics are considered. The primary one, which is the main focus of this work, is the percentage increase in manoeuvre duration relative to the GPOPS-II optimal time; being a relative quantity, it enables a consistent comparison across manoeuvres of varying angle and axis. The second metric is the per-step execution time, assessed in Section 4.2.2, which quantifies the computational load and thus the suitability of each controller for onboard deployment.

The controllers are first evaluated under nominal conditions and then progressively subjected to more demanding scenarios, in order to assess their robustness. Each of the following sections presents one such test, together with its specific setup and the corresponding results.

4.2. Test results – Nominal conditions

The controllers are first evaluated under nominal conditions, that is, with the uniformly distributed 12.0 kg mass of the reference 6U CubeSat introduced in Section 2. They are assessed along the two metrics defined in Section 4.1: the increase in manoeuvre time relative to the GPOPS-II optimal solution, which measures the optimality gap, and the per-step execution time, which governs the suitability for onboard deployment. The two aspects are examined in turn below.

Table 1
Performance comparison across architectures, in terms of percentage increase in manoeuvre duration relative to the optimal time computed by GPOPS-II.

Act Fun	Arch	Mean [%]	Std [%]	P90 [%]	Failed
ReLU	Regressor	0.97	4.51	6.68	0
	BinClass	2.67	4.89	9.21	0
	8Class	3.06	5.03	10.18	0
	ClMO	2.54	4.98	9.25	0
Softplus	Regressor	1.00	5.08	5.87	16 (0.09%)
	BinClass	1.87	4.97	8.09	0
	8Class	3.00	6.02	9.94	1 (0.01%)
	ClMO	1.96	5.66	8.81	0
SIREN	Regressor	0.62	5.09	4.61	55 (0.30%)
	BinClass	1.59	5.49	6.90	38 (0.21%)
	8Class	1.84	5.64	7.72	85 (0.46%)
	ClMO	1.33	6.09	6.68	51 (0.28%)
RL v1		2.59	4.45	8.93	0
RL v2		1.72	2.48	4.23	0
NMPC[a]		48.29	23.46	85.77	1 (0.01%)
IDVD	5th–50pt	4.58	1.30	6.29	0
	5th–100pt	4.60	1.30	6.31	0
	7th–50pt	2.62	0.95	3.78	0
	7th–100pt	2.62	0.96	3.79	0

[a] For the NMPC, owing to its substantially larger duration increases, a manoeuvre is classified as a failure when its duration exceeds an absolute value of 60 s, rather than the 100% relative increase adopted for the other methods (see Section 4.1).

4.2.1. Manoeuvre-time

Table 1 presents the manoeuvre-time results of the nominal simulation campaign for the proposed G&CNET architectures and for the RL, NMPC, and IDVD baselines. The table reports mean, standard deviation, and 90th percentile (P90) values, excluding failed manoeuvres. The "Failed" column reports both the absolute and the relative (out of 18,432 total) count of unsuccessful manoeuvres, defined for the G&CNETs and the RL agents as those whose duration exceeds the GPOPS-II optimal time by more than 100%, and for the NMPC as those exceeding an absolute duration of 60 s (see Section 4.1).

Despite good training performance and low average percentage increases, models using sinusoidal activations exhibited the highest number of failures. This is an interesting result as SIREN-based neural networks have been shown to perform particularly well in other applications [58]. Conversely, all ReLU-based architectures completed every manoeuvre successfully. The Softplus activation failed only 16 times (0.09%) with the Regressor and once (0.01%) with the 8-Class Classifier. All architectures achieved mean overheads of at most 3.06%. While standard deviations between 4.5% and 6.1% indicate occasional suboptimal performance, the P90 values confirm that 90% of the manoeuvres stayed within 10.18% of the optimal duration.

The two RL agents achieved mean percentage increases comparable to those of the proposed networks (2.59% and 1.72%), whereas the NMPC exhibited substantially higher increases. For the RL agents, although the time overhead remains well within acceptable limits, strict time-optimality cannot be mathematically guaranteed, as the control policy is entirely governed by the reward function design. Notably, however, both RL agents successfully completed all manoeuvres without a single failure.

Regarding the NMPC approach, the increase in manoeuvre time is substantial. An analysis of the NMPC control profiles reveals them to be significantly smoother than those generated by the RL agent and our proposed architectures, which are inherently bang–bang (or bang-zero-bang) by design. This residual overhead is intrinsic to the smooth, quadratic-cost nature of the controller rather than a symptom of poor tuning: a dedicated cost-weight and horizon sensitivity study (see Appendix B) shows that the adopted weights already lie in the fastest, cleanly-convergent region of the parameter space, and that pushing the state weights further does not yield a genuine reduction

Table 2
Mean inference time for all the presented models measured on different platforms: an `aarch64` laptop (PC), Raspberry Pi 4 and 5, and NVIDIA Jetson Orin AGX. For the Jetson two values are reported: on the left the execution on the CPU of the board, as for all the other platforms, and on the right the execution on its GPU. No GPU value is reported for the NMPC, since acados does not provide a GPU backend.

Act Fun	Arch	PC [ms]	Pi4 [ms]	Pi5 [ms]	Jetson [ms]
ReLU	Regressor	0.007	0.166	0.035	0.046/0.101
	BinClass	0.008	0.169	0.035	0.046/0.095
	8Class	0.008	0.173	0.037	0.048/0.101
	ClMO	0.009	0.203	0.043	0.055/0.146
Softplus	Regressor	0.014	0.204	0.058	0.055/0.114
	BinClass	0.014	0.203	0.057	0.055/0.110
	8Class	0.015	0.209	0.059	0.056/0.117
	ClMO	0.016	0.239	0.065	0.063/0.159
SIREN	Regressor	0.010	0.198	0.043	0.055/0.111
	BinClass	0.010	0.197	0.042	0.056/0.117
	8Class	0.009	0.204	0.043	0.056/0.115
	ClMO	0.011	0.234	0.050	0.063/0.156
RL (v1)		0.009	0.189	0.038	0.050/0.109
RL (v2)		0.008	0.190	0.039	0.050/0.111
NMPC		0.245	2.380	0.818	0.875/–

in manoeuvre time. Consequently, although the NMPC exhibits the largest manoeuvre-time increase among the compared methods, under the adopted 60 s failure criterion it converged on all but a single manoeuvre, the maximum recorded convergence time being 39.10 s.

Finally, for further reference, Table 1 includes results from the Inverse Dynamics in the Virtual Domain (IDVD) method [34,35]. Although IDVD exhibits a tighter distribution, the proposed neural networks achieve shorter manoeuvre times overall. It is important to remark that learning-based approaches, such as G&CNETs and RL, present a fundamental operational difference compared to IDVD. While IDVD functions strictly as a guidance algorithm that requires solving an optimisation problem to generate complete trajectories upfront, both the neural architectures and the RL agent deliver integrated guidance *and* control in a rapid, step-by-step fashion, thereby entirely bypassing the computational burden of online optimisation.

4.2.2. Inference time

The second key metric is the mean inference time per control step. This metric reflects the computational load of each network and is critical for assessing deployability on embedded platforms. All the controllers were exported to the ONNX format and evaluated through ONNX Runtime, so that the very same computational graph is executed on every platform and the comparison is not affected by the framework in which each model was originally trained. Inference times were measured on a laptop with `aarch64` architecture and on two commercial embedded platforms (Raspberry Pi 4 and 5), all CPU-only and without GPU acceleration. Table 2 reports the inference times, averaged over all simulations. On the Raspberry Pi 5 the networks required between 0.035 and 0.065 ms per control step, against the 0.007–0.016 ms measured on the laptop, whereas the Raspberry Pi 4, the oldest board considered, stayed below 0.24 ms. Even on the slowest platform the control action is thus produced in a time more than two orders of magnitude shorter than the 0.05 s control interval adopted here. These results confirm that network execution would not be the limiting factor for real-time deployment, with sensor dynamics likely imposing stricter timing constraints.

Although this work focuses on CPU-only platforms, GPU-enabled embedded boards are now widely available as commercial off-the-shelf components, with growing space qualification efforts driven by the increasing adoption of artificial intelligence techniques. For completeness, we thus report results from tests conducted on an NVIDIA Jetson Orin AGX, running the same networks both on the CPU and on the GPU of the board. As might be expected, the GPU does not accelerate the inference: on the same board it is about twice as slow as the CPU. The full potential of a graphics accelerator comes mainly from processing many samples at once, which is not possible for a feedback controller, since it must evaluate a single state and wait for the corresponding torque before the next state is even known. The fixed cost of handing the work to the accelerator and retrieving the result therefore remains dominant and, for a network with only seven input features (quaternion error components plus angular velocity error components), it exceeds the cost of the arithmetic itself, which a CPU carries out just as easily.

Regarding the baseline methods, it should be noted that they were specifically selected as state-of-the-art approaches capable of performing well on computationally limited platforms. When executed through the same runtime, the RL agents and the proposed models require comparable times, both reducing to a single forward pass through a small multilayer perceptron. The NMPC, in contrast, requires more than an order of magnitude longer, since an optimisation problem has to be solved at every control step. However, solving the nonlinear optimal control problem in 2.38 ms on a Raspberry Pi 4 still highlights the remarkable efficiency of the acados solver, especially considering the strong nonlinearity and complexity of the dynamics underlying the time-optimal reorientation of an asymmetric rigid body, which would normally be expected to entail significantly longer optimisation times.

Finally, as previously discussed, the IDVD method operates primarily as an open-loop guidance strategy, requiring the full trajectory to be generated in advance rather than computing control actions step by step online. Owing to this fundamental operational difference, a direct comparison of computation times with the methods presented in this work is not meaningful. Readers are referred to Ventura et al. [35] for a detailed performance analysis of the IDVD approach.

4.3. Test results – Tighter attitude accuracy

The controllers are next evaluated under a tighter attitude accuracy requirement, reducing the termination threshold on the attitude error from 0.5 deg to 0.1 deg (0.00175 rad), while the angular velocity tolerance is kept at its nominal value of 0.5 deg/s. All other settings are left unchanged.

Fig. 4 reports the mean percentage increase in manoeuvre time for all the considered methods under the two accuracy levels (solid bars: nominal 0.5 deg; hatched bars: tighter 0.1 deg). Focusing on the G&CNETs, tightening the requirement increases the manoeuvre time across all architectures, since additional time is needed to drive the residual attitude error below the stricter threshold. The Regressor remains the best-performing architecture for every activation, while the ordering of the remaining three depends on the activation function. Averaged over the three activations, the Regressor is followed by the Binary Classifier, the Class Multi Output and, last, the 8-Class Classifier. The failure rate remains modest for most architectures (below 2.4%), the only notable exception being the Class Multi Output with Softplus activation, which fails 12.3% of the manoeuvres.

The two baseline methods respond very differently to the tighter requirement. The RL agents, having been trained with a 0.5 deg termination condition, degrade markedly, failing 40.7% and 54.7% of the manoeuvres, respectively: not having been trained to reach such a fine accuracy, they struggle to null the residual error within the allotted time. The NMPC, in contrast, is far less affected: its mean manoeuvre-time increase grows from about 48% to 73%, while convergence remains essentially unchanged, with a single failed manoeuvre and a maximum recorded convergence time of 41.60 s.

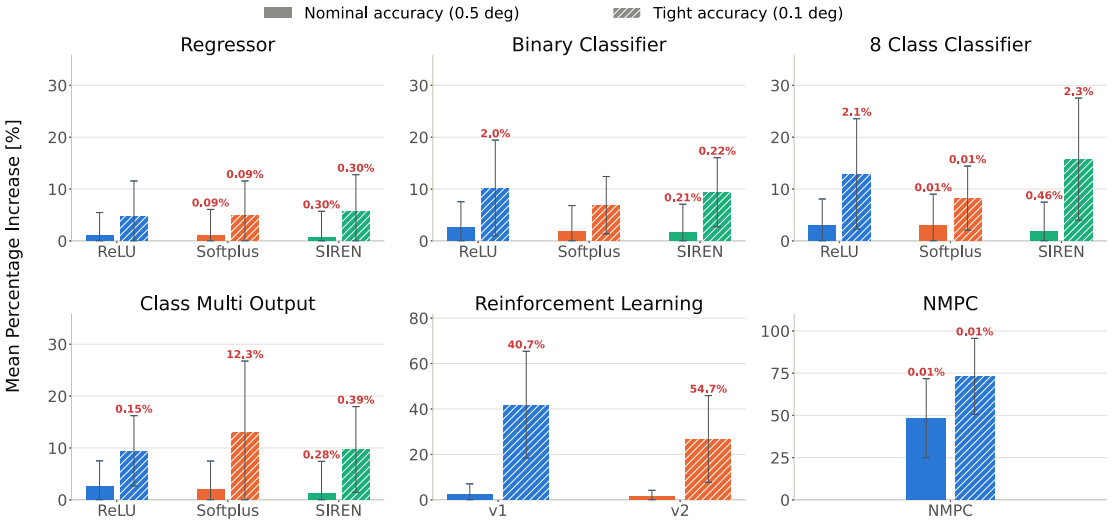


Fig. 4. Mean percentage increase in manoeuvre time relative to the GPOPS-II optimal solution, under the nominal 0.5 deg attitude accuracy (solid bars) and the tighter 0.1 deg accuracy (hatched bars). The first four panels report the G&CNET architectures, with one pair of bars per activation (ReLU, Softplus, SIREN), while the last two panels report the RL agents and the NMPC. Error bars denote the standard deviation (lower arm truncated at zero); the failure rate is annotated in red above every bar with at least one failed manoeuvre. Note the different vertical scales of the G&CNET, RL, and NMPC groups.

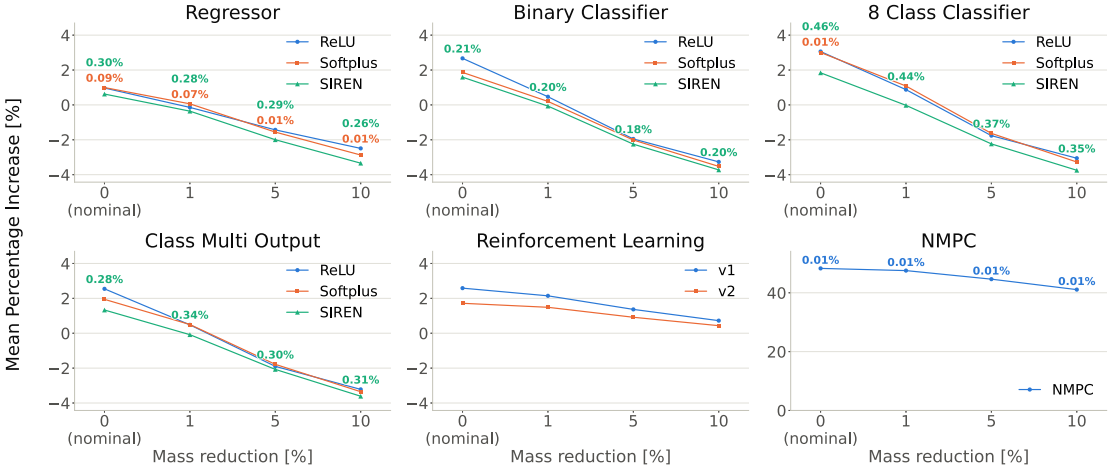


Fig. 5. Mean percentage increase in manoeuvre time relative to the GPOPS-II optimal solution as a function of the mass reduction (0% nominal, 1%, 5%, 10%). The first four panels report the G&CNET architectures (lines distinguished by colour and marker: ReLU, circles; Softplus, squares; SIREN, triangles), while the last two panels report the RL agents and the NMPC. The failure rate is annotated, in each curve's own colour, above every point where at least one manoeuvre fails; note the different vertical scale of the NMPC panel.

4.4. Test results – Reduced mass

The controllers are now evaluated under a *reduced mass*, still uniformly distributed, so that the elements of the inertia matrix are uniformly scaled down with respect to their nominal values. Mass reductions of 1%, 5% and 10% are analysed, while the termination conditions are kept at their nominal values of 0.5 deg attitude accuracy and 0.5 deg/s angular velocity accuracy. All other settings are left unchanged.

Fig. 5 shows the percentage increase in manoeuvre time relative to the GPOPS-II optimal solution, across all methods and mass reduction levels. The leftmost point of each curve represents the nominal case (12.0 kg). Notably, the percentage increase *decreases* as the mass is reduced. Since the available control torque is unchanged, it is more effective at the lower inertia, and the manoeuvre is completed in less time than the nominal-mass optimum. The key result is the networks' robustness. Trained exclusively on nominal-mass trajectories, they handle every reduced-mass scenario, with failure rates never exceeding 0.5%, demonstrating generalisation to realistic inertia mismatches such as those induced by gradual fuel consumption.

Both baseline methods exhibit the same decreasing trend: the RL agents remain failure-free across all mass reduction levels, and the NMPC likewise converges on essentially all manoeuvres, its mean increase decreasing from about 48% in the nominal case to 41% at the highest (10%) mass reduction.

4.5. Test results – Non-uniform mass distribution

The mass is now no longer uniformly distributed, so that the principal axes of inertia no longer coincide with the geometric body axes. The inertia matrix with respect to the body frame is therefore obtained by rotating the nominal diagonal inertia matrix, originally expressed in the principal-axes frame, by means of the standard inertia-tensor frame transformation [79]

$$^{\mathcal{B}}[I] = C_{\mathcal{BP}}\, ^{\mathcal{P}}[I]\, C_{\mathcal{BP}}^{T}, \tag{6}$$

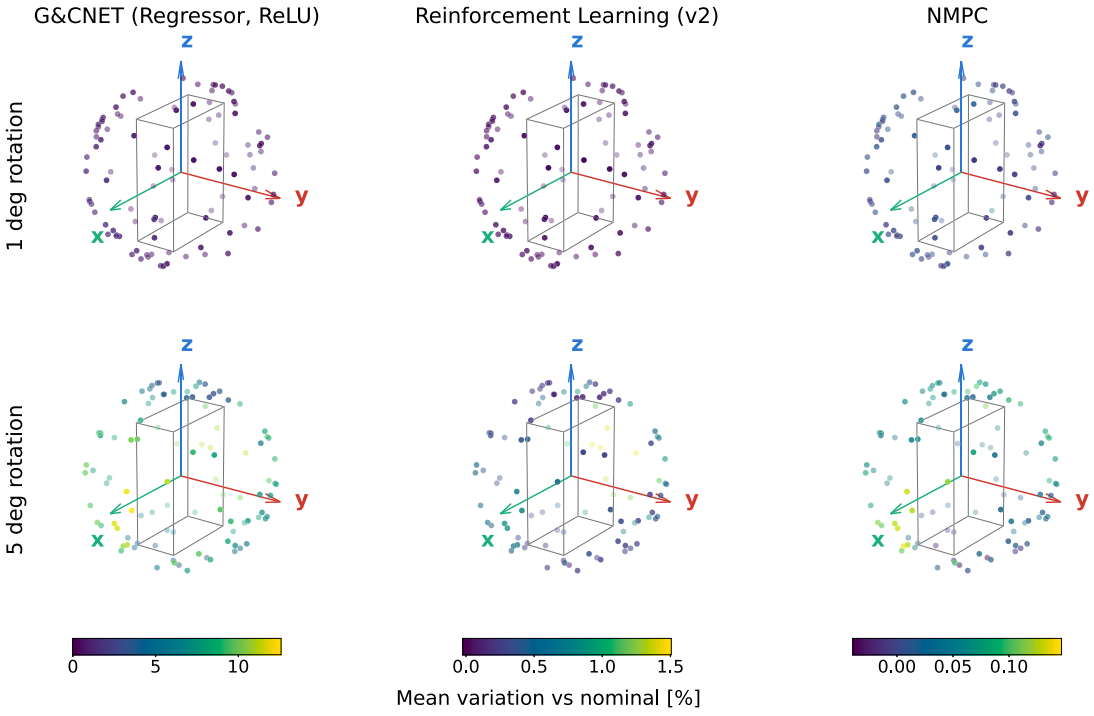


Fig. 6. Performance under non-diagonal inertia matrices for the best model of each method (columns: G&CNET ReLU-Regressor, RL v2, NMPC) and for the two principal-to-body misalignments (rows: 1 deg, 5 deg). Each point corresponds to one inertia matrix and is placed at the perturbation rotation axis on the unit sphere; its colour encodes the mean percentage variation in manoeuvre time relative to the nominal (diagonal-inertia) case, over the 3072 evaluated manoeuvres. Note the per-method colour scale. The labelled arrows denote the spacecraft body x-, y-, and z-axes, and the grey box the 6U spacecraft. (For a correct interpretation of the colour map, the reader is referred to the web version of this article.)

where $C_{\mathcal{BP}}$ is the direction cosine matrix (DCM) of the principal-axes frame $\mathcal{P}$ relative to the geometric body frame $\mathcal{B}$, ${}^{\mathcal{P}}[I]$ is the nominal diagonal inertia matrix expressed in the principal-axes frame, and ${}^{\mathcal{B}}[I]$ is the resulting inertia matrix in the body frame, which includes small off-diagonal terms.

To probe this mismatch, we sample 100 rotation axes uniformly on the unit sphere and, for each, apply a fixed rotation of 1 deg and of 5 deg to the principal-axes frame, producing 200 non-diagonal inertia matrices in total (100 per angle). For each method we test the best-performing model identified in the previous scenarios, i.e. the ReLU-Regressor for the G&CNETs and the v2 agent for RL, so as to compare the three approaches on an equal footing under inertia uncertainty. To keep the computational cost tractable while retaining full and uniform coverage of the manoeuvre space, each matrix is evaluated on a subset of 3072 manoeuvres, obtained by drawing one manoeuvre per equal-area cell of the test-set sphere; this preserves a uniform distribution over both the rotation axis and the rotation angle. All simulations adopt the nominal termination conditions (0.5 deg attitude, 0.5 deg/s angular velocity) and the same failure criteria used throughout: a manoeuvre-time increase above 100% for the G&CNETs and RL, and an absolute duration above 60 s for the NMPC.

Fig. 6 reports the results. Most notably, *no manoeuvre fails* for any of the three methods across all 200 matrices, confirming a strong robustness to inertia mismatch even though the G&CNETs and the RL agent were trained solely on the nominal, diagonal inertia. The mean manoeuvre-time increase relative to the GPOPS-II optimal solution grows with the misalignment: for the ReLU-Regressor it rises from 1.0% in the nominal case to 1.7% at 1 deg and 8.4% at 5 deg, the worst matrix reaching 13.6%. The RL agent, whose nominal increase is 1.7%, only reaches 2.2% at 5 deg, while the NMPC, dominated by its intrinsic overhead, stays at 48.2% against a nominal 48.1%.

Measured as the degradation caused by the perturbation alone, that is the difference with respect to the nominal case shown in Fig. 6, the three methods differ by more than an order of magnitude: at 5 deg the ReLU-Regressor loses 7.4 percentage points, the RL agent 0.5 and the NMPC 0.05. The sensitivity of the G&CNETs to an inertia mismatch is therefore substantially higher than that of the two baselines, although it never compromises the completion of the manoeuvre. The degradation also exhibits a clear and consistent directional dependence: it is largest when the perturbation axis is aligned with the geometric x-axis and smallest when aligned with the z-axis. For the ReLU-Regressor the degradation for x- versus z-dominated rotations is 0.9 against 0.4 percentage points at 1 deg, and 10.3 against 5.3 at 5 deg; the same ordering, confirmed by a positive correlation between the degradation and the x-component of the perturbation axis, holds for all three methods and at both rotation magnitudes. This reflects the spacecraft geometry: rotations about the x-axis introduce the largest products of inertia, inducing stronger gyroscopic cross-coupling and hence a larger departure from the nominal behaviour.

4.6. Test results – Sensor noise

The tests presented so far assumed perfect state knowledge. In this section, the robustness of the proposed controllers to measurement noise is assessed by corrupting the state provided to the network at each control step, while the spacecraft dynamics are propagated using the true state. Attitude measurement noise is modelled as a small random rotation superimposed on the true attitude. To this end, at each control step, a perturbation quaternion δq is built from a rotation axis $\hat{\mathbf{e}}$ drawn uniformly on the unit sphere and a rotation angle $\delta\phi \sim \mathcal{N}(0, \sigma_{att}^2)$, and it is composed with the true attitude quaternion. The angular velocity noise is instead additive and is applied independently to each component as $\boldsymbol{\omega}_{meas} = \boldsymbol{\omega}_{true} + \boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma_{rate}^2 \mathbf{I}_3)$. The corrupted measurements are then used to compute the state error $\mathbf{x}_{err}$ that is fed to the network. Three noise levels are considered, whose values are selected as fractions of the termination conditions introduced in

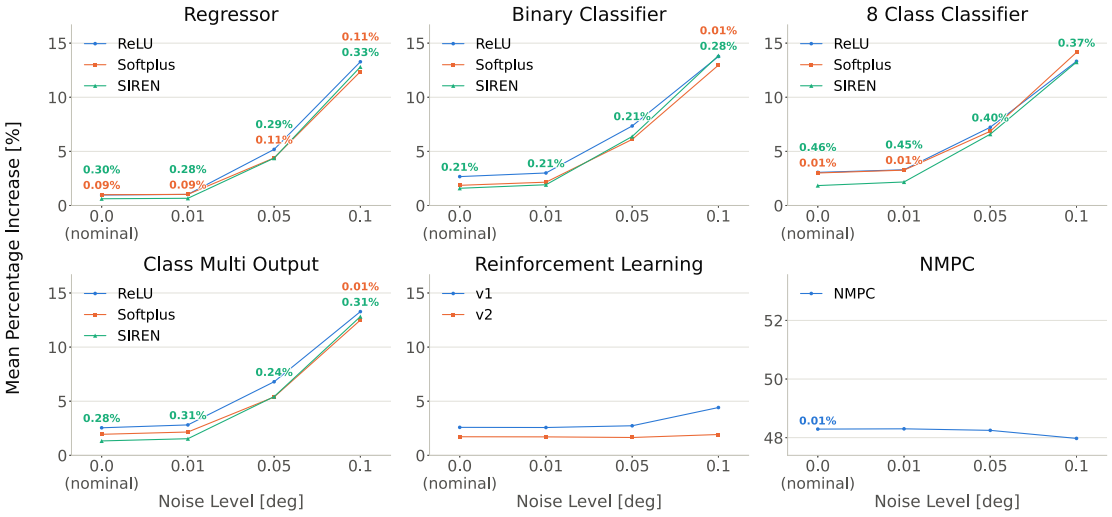


Fig. 7. Mean percentage increase in manoeuvre time relative to the GPOPS-II optimal solution as a function of the injected sensor-noise level (nominal, 0.01, 0.05, 0.1 deg). The first four panels report the G&CNET architectures (lines distinguished by colour and marker: ReLU, circles; Softplus, squares; SIREN, triangles), the last two the RL agents and the NMPC. The failure rate is annotated, in each curve's own colour, above every point where at least one manoeuvre fails; note the different vertical scale of the NMPC panel.

Section 4.1 (0.5 deg attitude accuracy and 0.5 deg/s angular velocity accuracy). The two standard deviations are set to the same numerical value at each level, namely $\sigma_{att} = 0.01$, 0.05 and 0.10 deg with $\sigma_{rate} = 0.01$, 0.05 and 0.10 deg/s, corresponding to 2%, 10% and 20% of the termination thresholds. The same three levels are also adopted for the state estimation error campaign of Section 4.7.

The full test campaign described in Section 4.1 is repeated for each noise level, with an independent noise realisation drawn for each of the 18,432 manoeuvres. Fig. 7 reports the mean percentage increase in manoeuvre time as a function of the injected noise level, with the nominal (noise-free) case included as the leftmost point of each curve. Performance degrades progressively and smoothly with increasing noise magnitude, without any abrupt loss of effectiveness up to the highest tested level: for the best-performing architecture (the ReLU-Regressor) the mean increase grows from 0.97% in the nominal case to 13.28% at the highest level, and all G&CNET architectures remain within a few percentage points of one another. This degradation, however, is not accompanied by a loss of reliability: the failure rate stays below 0.15% for all ReLU- and Softplus-based architectures, while for the SIREN-based networks it fluctuates around the small fraction already observed under nominal conditions, without any systematic noise-induced trend. The two baseline methods are markedly insensitive to this type of noise: the RL agents grow by at most a few percent and never fail, and the NMPC mean increase is virtually unchanged (from 48.3% in the nominal case to 48.0% at the highest level).

4.7. Test results – State estimation error

Measurement noise is not the only source of navigation error affecting an onboard controller. In a realistic implementation, the state provided to the controller is the output of an attitude estimator, whose residual error is correlated in time and, over the short duration of a slew manoeuvre, is more appropriately represented as a slowly varying offset than as white noise. Testing both cases therefore brackets any intermediate degree of time correlation.

Accordingly, a constant navigation error is considered here. At the beginning of each manoeuvre, a perturbation quaternion δq is built from a rotation axis $\hat{\mathbf{e}}$ drawn uniformly on the unit sphere and a rotation angle $\delta\phi \sim \mathcal{N}(0, \sigma_{att}^2)$, and it is composed with the true attitude quaternion, while a constant angular velocity offset $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma_{rate}^2 \mathbf{I}_3)$ is added to the true angular velocity. Unlike the previous case, both offsets are drawn once at the beginning of the manoeuvre and then held constant, and they are resampled at each run, so that the reported statistics account for all possible error directions with respect to the spacecraft body axes. The same three magnitude levels adopted in Section 4.6 are used, allowing a direct comparison between the two error models at equal standard deviation.

A constant offset is particularly critical for rest-to-rest manoeuvres. Acting on a corrupted state, the controller does not drive the spacecraft towards the intended target, but towards the one it perceives. The two components of the offset differ in this respect: a constant attitude offset merely displaces the equilibrium, so that the spacecraft settles on an attitude close to, but distinct from, the desired one; a constant angular-velocity offset instead leaves the closed loop without any equilibrium, since the controller perceives the spacecraft as being at rest only while it is in fact still rotating at $-\boldsymbol{\eta}$, so that the true attitude keeps drifting and the controller has to re-engage indefinitely. This distinction between the true and the perceived state also defines the evaluation criterion: the manoeuvre is terminated on the basis of the true state, while the controller only receives the corrupted one, so that what is assessed is the controller's ability to reach the true target while acting on a corrupted state error.

Fig. 8 reports the mean percentage increase in manoeuvre time as a function of the offset magnitude. Two observations stand out when comparing this scenario with the uncorrelated one of Section 4.6 at equal standard deviation. First, in terms of *mean* manoeuvre time the degradation is actually milder: for the ReLU-Regressor the mean increase reaches only 4.54% at the highest level, against 13.28% under white noise, since a constant offset does not continuously perturb the commanded torque. Second, and more importantly, the constant offset is nonetheless the more demanding scenario, because it is the only one that induces genuine failures. The reason is the one discussed above: the zero-mean noise of the previous campaign perturbs the commanded torque continuously but leaves the rest condition attainable, whereas a constant angular-velocity offset is incompatible with it, while the attitude offset further displaces the point about which the spacecraft settles, so that convergence relies on the true state crossing the acceptance region while the attitude drifts. At the highest tested level these failures are mostly registered on the two baseline methods: the RL agents fail on 4.3% (v1) and 2.0% (v2) of the manoeuvres and the NMPC on 4.1%, whereas the G&CNET architectures remain essentially failure-free, with failure rates below 0.5% for every architecture–activation combination.

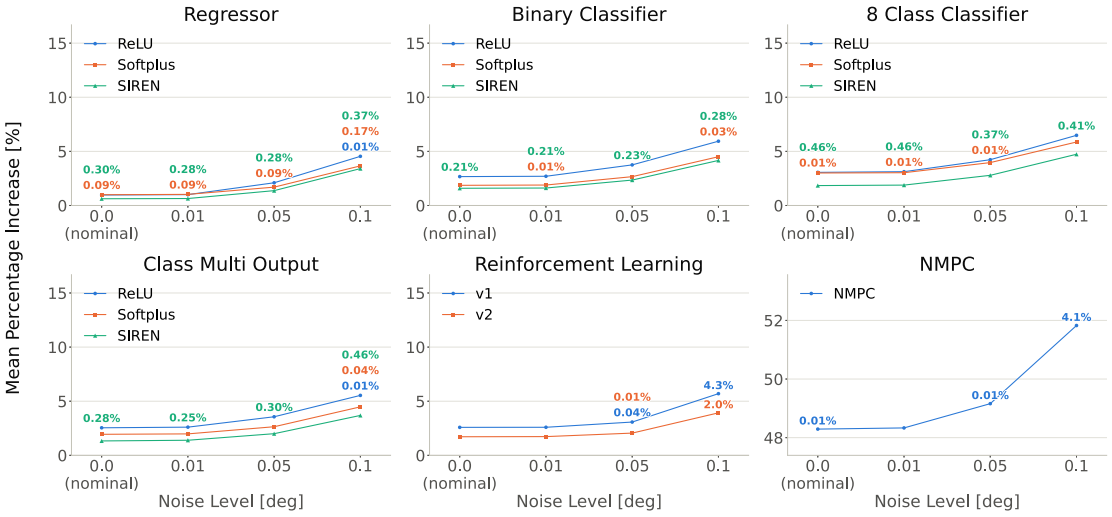


Fig. 8. Mean percentage increase in manoeuvre time relative to the GPOPS-II optimal solution as a function of the constant navigation-offset magnitude (nominal, 0.01, 0.05, 0.1 deg). The first four panels report the G&CNET architectures (lines distinguished by colour and marker: ReLU, circles; Softplus, squares; SIREN, triangles), the last two the RL agents and the NMPC. The failure rate is annotated, in each curve's own colour, above every point where at least one manoeuvre fails; note the different vertical scale of the NMPC panel.

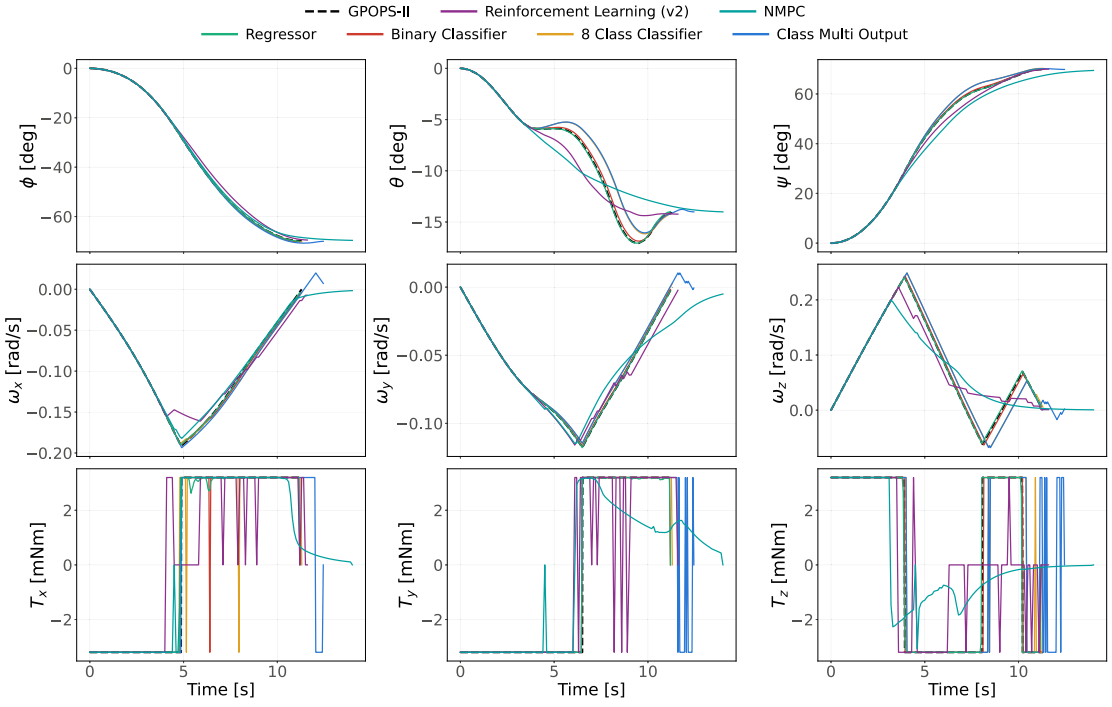


Fig. 9. Comparison of attitude (parametrised as 321 Euler angles), angular velocity, and control torque profiles for a representative 90 deg rotation about a non-principal (body-diagonal) axis, for the GPOPS-II time-optimal reference, the four G&CNET architectures, the RL agent (v2), and the NMPC. The neural-network-based controllers closely reproduce the GPOPS-II reference trajectory; classification-based models yield bang–bang control actions, and the regressor, despite its continuous output, remains saturated for nearly the whole manoeuvre. The RL agent converges to the same target with comparable state profiles, but its commanded torque switches repeatedly and passes through intervals of null torque. The NMPC instead tracks the reference with visibly smoother, chattering-free control profiles, at the cost of a longer manoeuvre time. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Overall, the results of both campaigns indicate that the proposed controllers tolerate navigation errors up to a magnitude comparable to the required pointing accuracy, with monotonic degradation of the manoeuvre time and without catastrophic loss of stability. Across both error models and all the tested levels, no architecture–activation combination exceeds a failure rate of 0.5%, whereas both baselines lose reliability as soon as the navigation error becomes correlated in time.

4.8. Test results – Trajectory profiles

Fig. 9 provides a final qualitative comparison of the attitude, angular velocity, and control torque profiles for a representative 90-degree rotation about a non-principal (body-diagonal) axis. This example illustrates the close correspondence between the neural network outputs and the GPOPS-II time-optimal reference. The classification-based

architectures generate bang–bang control actions, and the regressor behaves in the same way despite its continuous output: it departs from saturation only in the immediate vicinity of the target. Owing to these similarities in control behaviour, the resulting attitude trajectories closely follow the time-optimal solution. The RL agent (v2) reaches the same terminal attitude, and its attitude and angular-velocity profiles remain close to the reference, but its commanded torque departs from the single-switch profile of the expert, alternating repeatedly between the saturation levels and passing through intervals of null torque. The NMPC, in contrast, is the only controller whose commands are genuinely smooth and free of chattering, since its quadratic cost makes use of intermediate torque levels rather than of the saturation values alone; this comes at the price of a substantially longer manoeuvre.

4.8.1. Control smoothness and actuator usage

Although the main focus of this work is the manoeuvre time, the profiles make it worth adding a consideration on the energy and propellant expenditure of the compared methods. On the reduced test campaign of Section 4.5, the learning-based controllers (G&CNETs and RL agent) are found to operate at or near saturation for essentially the whole manoeuvre, being bang–bang by construction: the commanded torque is saturated for at least 98.7% of the time. Their energy and propellant consumption are therefore maximal and, to first order, simply proportional to the manoeuvre time. Even the ClassMultiOutput head, which can command a null torque, and the Regressor, whose output is continuous, exploit intermediate or zero actions in less than 1% of the time steps, that is, only in the immediate vicinity of the target. For this family of controllers, actuator-usage and energy metrics therefore add no discriminating dimension beyond the manoeuvre time, which is precisely why the latter is adopted as the primary figure of merit throughout this work.

The NMPC baseline, on the contrary, is the only method that departs from this behaviour. Its quadratic cost yields smoother commands that saturate only 63% of the time, translating into a lower energy consumption than the G&CNETs, at the price of a 48.7% longer manoeuvre. The NMPC thus embodies a clear smoothness/energy-versus-time trade-off: when the propellant budget or the actuator wear is the binding constraint and the time budget is relaxed, the smoother NMPC is advantageous, whereas the near-time-optimal G&CNETs maximise speed at the cost of continuous saturation.

5. Stability analysis

The lack of formal stability guarantees is a recognised obstacle to the use of learned controllers in safety-critical applications: a network maps the state to the control action through a composition of nonlinear layers, to which the classical techniques of formal stability analysis are typically not applicable. Several strategies have been proposed that train the controller and a neural Lyapunov function at the same time, so that a stability certificate is available by construction: Li et al. [83], for example, jointly synthesise a controller and a Lyapunov function for continuous-time systems, while Wang et al. [84] jointly supervise a neural candidate Lyapunov function and a control policy to obtain a certifiably stable, time- and fuel-optimal guidance law for spacecraft close-proximity operations. Even in this favourable setting, however, the certificate can be formally verified only in low dimension: in [83] verification is already challenging on six-dimensional benchmarks and fails on a twelve-dimensional one, so that the guarantee is ultimately supported by sampling a large number of closed-loop trajectories and confirming that they all converge. Sampling-based validation of this kind is what the literature falls back on once the dimension of the system makes formal verification impractical.

The controllers considered here are in a less favourable position: they are obtained by behavioural cloning of a time-optimal expert, so that no stability requirement enters the training procedure and no certificate is available by construction. Moreover, as the closed-loop state is seven-dimensional and the dynamics carry the gyroscopic coupling of a fully asymmetric body, establishing stability is therefore harder than in the certified setting above, and the assessment is accordingly carried out empirically, over the entire set of manoeuvres already simulated. This section first evaluates a Lyapunov candidate along the whole test campaign, for every architecture and activation function; it then reports the two formal verification routes that were attempted and why neither scales to the present problem. The last part of the section investigates the reason behind the instability observed for the SIREN-based models, and in particular behind the higher number of failures they record in the test campaign.

5.1. Empirical Lyapunov verification

We adopt a classical quaternion-based Lyapunov candidate function [85]

$$V(\boldsymbol{\omega}, q_{0,e}) = \frac{1}{2}\,\boldsymbol{\omega}^{\mathsf{T}}\mathbf{J}\boldsymbol{\omega} + K_p\left(1 - |q_{0,e}|\right), \tag{7}$$

where $\boldsymbol{\omega}$ is the spacecraft angular velocity, $\mathbf{J}$ the true inertia matrix, $q_e = [q_{0,e}, \boldsymbol{q}_{v,e}]^{\mathsf{T}}$ the attitude error quaternion defined in Section 2, and K_p a positive weighting constant, set to unity throughout. The first term is the rotational kinetic energy and the second a measure of the residual attitude error, so that V vanishes only when the spacecraft is at rest at the target, consistently with the rest-to-rest manoeuvres considered here. Differentiating along the dynamics of Eq. (1) and the kinematics of Eq. (2) gives

$$\dot{V} = \boldsymbol{\omega}^{\mathsf{T}}\mathbf{T} - \frac{1}{2}K_p\,\mathrm{sgn}(q_{0,e})\,\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{q}_{v,e}, \tag{8}$$

the gyroscopic contribution vanishing by the scalar triple product identity. Both terms depend only on the state error and on the commanded torque $\mathbf{T}$, and are therefore available at every control step of a closed-loop simulation without any additional propagation.

For an analytical control law the feedback is designed so that substituting it into Eq. (8) returns a negative semi-definite expression. This is not the case here, since the networks are trained to reproduce a time-optimal expert and not to satisfy $\dot{V} \leq 0$. Eqs. (7) and (8) are therefore used as a diagnostic rather than as a stability certificate, and is evaluated a posteriori on the test campaign already presented in Section 4: for each of the 18,432 nominal manoeuvres, and for every architecture and activation function of Section 3, $V(t)$ and $\dot{V}(t)$ are computed at each control step from the logged state error and commanded torque. What is of interest is where the condition $\dot{V} \leq 0$ is violated, by how much, and whether the violations compromise convergence.

Fig. 10 reports the amplitude of each recorded $\dot{V}(t) > 0$ peak against its normalised occurrence time within the manoeuvre, collected over all the 18,432 manoeuvres of the nominal test set. The picture is common to all controllers: violation instants are present for every architecture, the largest ones are clearly concentrated at the beginning of the manoeuvre, and in the final phase their amplitude decays as the target is approached: the median peak recorded in the last tenth of the manoeuvre is, however, one to two orders of magnitude smaller than in the first tenth, so that the terminal phase is essentially violation-free.

Rather than a controller defect, the violations in the early transient have a natural explanation in the theory of time-optimal reorientation. For a triaxial asymmetric body the minimum-time trajectory is not an eigenaxis rotation [7], that is, it is not the shortest-path manoeuvre along the geodesic. Since the controllers reproduce time-optimal manoeuvres, the path they follow is therefore typically not the shortest one towards the target, but a longer one that takes advantage of the gyroscopic coupling to reach the target in less time. Decomposing the Lyapunov function into its two terms shows that it is the attitude term, which measures the angular distance to the target, that dominates the initial rise: an angularly longer path makes V grow temporarily, so that $\dot{V} > 0$. A second, minor contribution comes from the kinetic

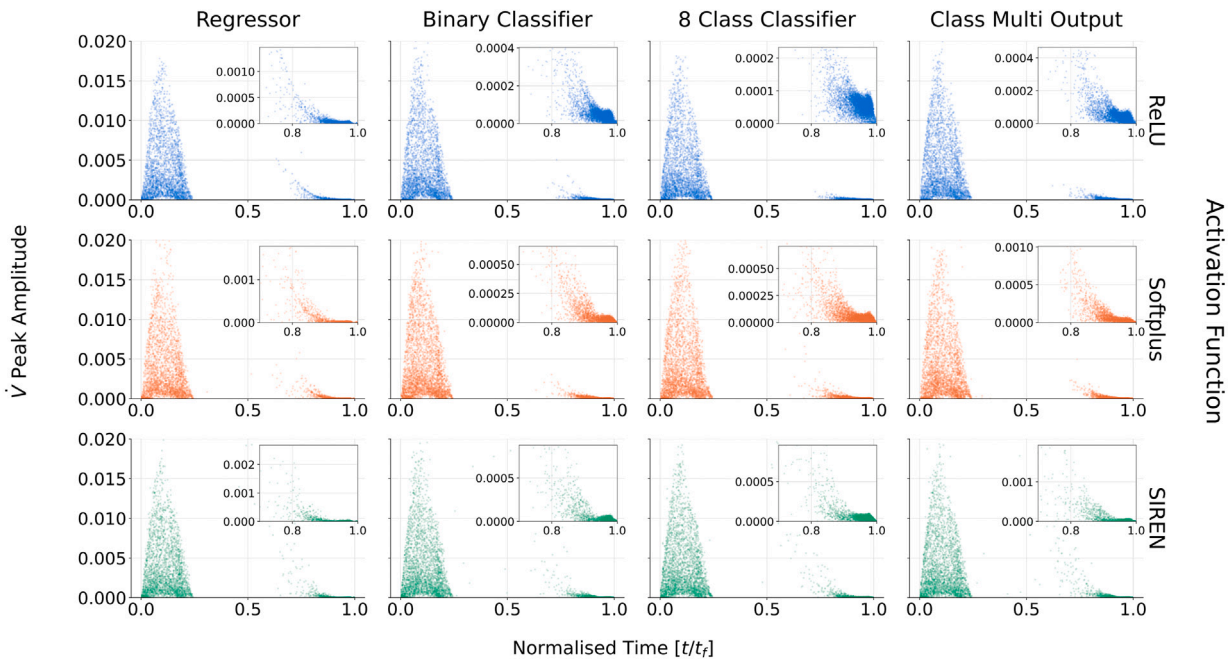


Fig. 10. Amplitude of each recorded Lyapunov violation ($\dot{V}(t) > 0$) against its normalised occurrence time t/t_f within the manoeuvre, for every G&CNET architecture (columns) and activation (rows). Each point is an individual violation peak. The inset zooms on the terminal phase ($t/t_f \geq 0.7$), where amplitudes are too small to be read on the main axes.

term, which necessarily grows while the spacecraft spins up from rest. Neither of the two involves the commanded torque, which enters $\dot{V}$ only through the power term $\boldsymbol{\omega}^{\mathsf{T}}\mathbf{T}$, positive when the control injects rotational energy instead of removing it. Decomposing the initial rise over a large sample of manoeuvres, spanning all architectures and activations, confirms this ranking: the attitude term accounts for the large majority of the increase, and the kinetic term for the remainder, consistently across every architecture–activation combination.

As a conclusive remark, it is important to recall that the violation of a Lyapunov condition does not imply a divergence of the trajectory, but simply tells us that in those regions the stability of the system cannot be certified by means of this Lyapunov function. Indeed, throughout Section 4.2 a manoeuvre is labelled as failed on a *performance* basis, that is when its settling time exceeds the adopted threshold, and not because the closed loop diverges. Adopting a more generous 100 s timeout for the simulation as the sole convergence criterion, the ReLU- and Softplus-based architectures reach the target on *every* manoeuvre (0 timeouts out of 18,432), and even the SIREN-based models genuinely fail to converge on only 0.01 to 0.15% of them, with several otherwise-failed manoeuvres settling as late as 99.6 s. The aspect is relevant for safety: a slow manoeuvre, which we may well classify as a failure, still drives the spacecraft to the desired attitude and does not leave it in a tumbling state, that is, an excessive settling time degrades the performance of the commanded reorientation but does not compromise the integrity of the mission. Only the small residual set of genuinely non-converging manoeuvres, confined to the SIREN activation, corresponds to a true dynamic failure, and the cause is investigated in the following.

5.2. Formal verification

Two routes towards a rigorous, rather than empirical, verification were also attempted. The first is a set-based reachability analysis of the closed-loop dynamics, in which the flow of the system is expanded as a high-order Taylor polynomial of the initial state by means of Differential Algebra (DA) [86,87], and an initial set of attitudes is propagated through the neural feedback law to bound the reachable set, with Automatic Domain Splitting (ADS) subdividing the domain wherever the expansion loses accuracy. The second is a direct verification of the network map through linear-relaxation bound propagation [88], which returns guaranteed bounds on the controller output over a bounded input set.

Neither route produced a usable result, and in both cases the reason is the curse of dimensionality. The closed-loop state is seven-dimensional and the dynamics are strongly nonlinear, and the flow has to be propagated through the hundreds of integration steps that make up a single manoeuvre: at every step the polynomial expansion loses accuracy and the domain has to be subdivided further, so that, at a fixed and computationally tractable expansion order, the number of ADS subdomains required to keep the truncation error bounded grows prohibitively. The bounds returned by the relaxation, in turn, become so wide after a few propagation steps that they no longer exclude any behaviour, and therefore certify nothing. This is the same limitation encountered in the works discussed above, where a verifier of the same family is applied to systems that are designed to be certifiable and is nonetheless replaced by the sampling of trajectories as soon as the dimension grows [83]. For this reason, in the present case as well, we resorted to an empirical verification.

5.3. Analysis of the closed-loop failures

From the empirical campaign of Section 4.2 it emerged that most of the failed manoeuvres originate from the controllers that employ the sinusoidal (SIREN) activation. Extending the simulation timeout to 100 s, the ReLU- and Softplus-based models reach the target on every one of the 18,432 manoeuvres, whereas the SIREN models retain a small residual set of unconverged manoeuvres: 21 manoeuvres (0.11%) for the Regressor, and between 2 and 27 for the classification heads. Because this set is small, the failing manoeuvres were inspected individually in order to identify the reason for the failure.

First, propagating the manoeuvres still unconverged at 100 s out to 2000 s shows that these are not ordinary slow convergences. Most of them (19 of the 21 for the Regressor) do eventually reach the target, but only after an extremely long and irregular settling, during which the attitude error keeps swinging over the full 0–180° range and the commanded torque remains in a persistent bang–bang chatter; two manoeuvres never converge at all. What distinguishes these cases is

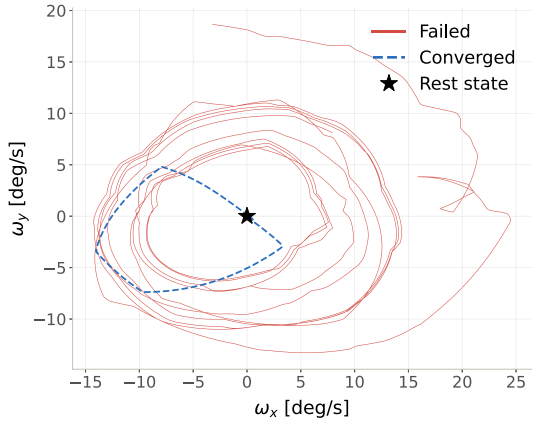


Fig. 11. Angular-velocity phase portrait of a representative failing SIREN manoeuvre (solid red line), which remains on a bounded set at non-zero angular velocity, compared with a manoeuvre that converges from a comparable rotation angle (dashed blue line) and spirals to rest (star at the origin). The initial transient is discarded for the failing trajectory.

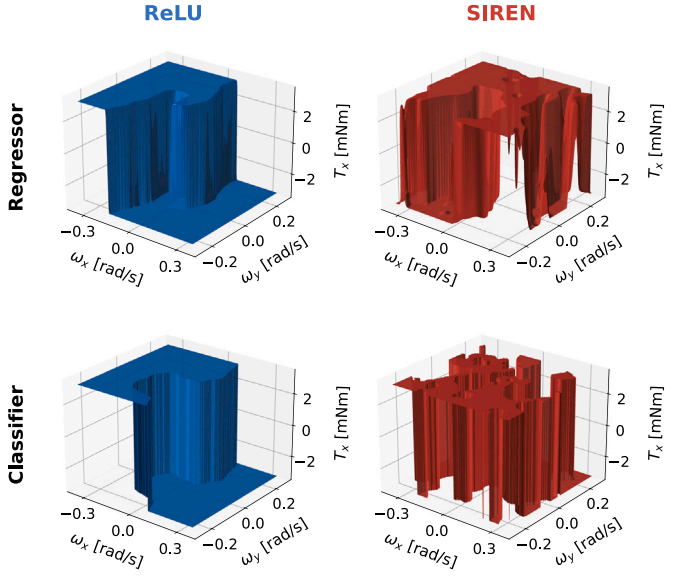


Fig. 12. Learned control surface $T_x(\omega_x, \omega_y)$ from true inference of the trained networks over the whole training domain. The error quaternion $\mathrm{q}_{\mathrm{err}}$ and the third angular-velocity component ω_z are held fixed at a failed-manoeuvre operating point, while ω_x and ω_y are swept; T_x is shown as a representative component, the same behaviour occurring for T_y and T_z. Columns: ReLU (left) vs SIREN (right); rows: continuous Regressor (top) vs discrete 8-class Classifier (bottom). ReLU realises the required bang–bang switch as a single clean transition, whereas the sinusoidal SIREN basis produces a pervasive Gibbs-like ringing (continuous case) or a fragmented decision boundary (discrete case).

best seen by comparing them with a manoeuvre that does converge from a comparable rotation angle, as in the angular-velocity phase portrait of Fig. 11, which contrasts a failing manoeuvre of 179.1 deg about the axis $[0.544,\ -0.059,\ 0.837]$ with a converging one of 172.7 deg about $[0.662,\ 0.410,\ -0.628]$: the converging manoeuvre spirals into the equilibrium, whereas the failing one remains on a bounded set at non-zero angular velocity. The failure mode is therefore not a slow approach to the target, but a sustained motion that does not approach it at all.

Second, it can be noticed that the failures are mostly localised at large rotation angles, which characterises the instability not as a generic property of the learned map, but as something triggered in a specific, marginal region. Two reasons make this the most demanding region of the manoeuvre space: first, it lies at the very edge of the training distribution, since the rotation angles of the data set extend only up to 180 deg; secondly, manoeuvres from this region require the largest terminal deceleration, because a wider rotation is executed at a higher peak angular velocity.

In order to identify the root cause we decided to examine the learned control surface itself. Since the time-optimal control law is bang–bang, each torque component only takes its two saturated values: for any given state the network has to output either the positive or the negative saturation level, and it has to switch abruptly from one to the other as the state evolves. The function it is required to reproduce is therefore not a smooth one but a sharp step, and how faithfully an architecture can represent a step becomes the relevant question. As the controller input $\mathbf{x}_{\mathrm{err}} = [\mathrm{q}_{\mathrm{err}}\ \boldsymbol{\omega}_{\mathrm{err}}]^\top$ is seven-dimensional, to render a torque component as a three-dimensional surface we had to fix five of the seven inputs. We fixed the four error-quaternion components and one angular-velocity component (ω_z) at values taken from an operating point on a failed SIREN trajectory and we swept the selected angular velocities (ω_x, ω_y) across the whole range seen in training. We choose to fix the ω_z component but sweeping any other pair yields the same qualitative picture. Fig. 12 shows the resulting surface for the T_x component, obtained by true inference of the trained networks; the same behaviour is found for the other two components, and one is displayed for clarity. The Regressor is compared with the 8-class Classifier, taken as its discrete counterpart. A ReLU network, being piecewise-linear, renders the required switch as a single clean transition (left panels). The sinusoidal basis of SIREN, by contrast, approximates the step with spurious oscillations, a Gibbs-like ringing, which pervade the whole domain (right panels). The effect is not an artefact of a single trajectory but a structural property of the activation: approximating a sharp step with a composition of sinusoidal functions characteristically brings with it spurious oscillations around it.

Analysing the trajectories, it can be noticed that the ringing has also a direct dynamical consequence. Where the control surface oscillates, the local sensitivity of the torque to the angular velocity, i.e. $\partial\mathbf{T}/\partial\boldsymbol{\omega}$, can acquire the wrong sign causing the commanded torque to momentarily reinforce the angular velocity instead of opposing it, thus injecting rotational energy. Measured along the trajectories, it can be noticed that the failing SIREN manoeuvres can spend about 32% of their time in an anti-damping condition, against only 2% for ReLU on the same manoeuvres. The ringing is a property of the whole learned surface, whereas the anti-damping is measured along a trajectory, which visits only a curve across that surface: not every such curve crosses the oscillations where they are strong enough to matter, and this is why the large majority of the manoeuvres still converge. They are not free of the effect, however: the SIREN manoeuvres that do converge still spend about 6% of their time in anti-damping, some three times the ReLU value, so the oscillations are met on them as well, only not long enough for the injected energy to prevail over the dissipation.

The two observations combine to explain why only a few manoeuvres fail. By itself, an anti-damping episode is not destructive: the energy it injects is removed by the subsequent, correctly damping commands, and this is precisely what happens on the manoeuvres that converge, where such episodes remain occasional. What decides the outcome is therefore the balance between the energy injected and the energy the controller removes. In the large-angle region identified above this balance degrades on both sides: the network operates at the very edge of its training distribution, so the anti-damping episodes become far more frequent (from about 6% to about 32% of the time) and, at the same time, the angular momentum to be removed is the largest of the whole manoeuvre space. When the injection ceases to be occasional, the dissipation no longer prevails, and, instead of spiralling into the equilibrium, the state keeps wandering on a bounded set

similar to the one shown in Fig. 11. The chain behind the failures is thus complete: the bang–bang expert requires the network to reproduce a discontinuous map, the sinusoidal basis approximates it with a persistent ringing that occasionally makes the torque reinforce the angular velocity instead of opposing it, and in that marginal region this happens often enough to prevent convergence. Notably, the cause is not a poor reproduction of the expert, as the SIREN models attained the lowest training and validation losses among the activations considered (Section 3), but the shape of the control surface through which they realise it.

6. Conclusion

This work investigated, for the first time, to the best of the authors' knowledge, the application of Guidance and Control Networks (G&CNETs) to the time-optimal spacecraft reorientation problem. This problem represents a particularly challenging benchmark for learning-based control methods due to the highly nonlinear rigid-body dynamics involved and the bang–bang nature of the underlying optimal policy.

The results demonstrated that lightweight G&CNET architectures are capable of accurately approximating the optimal state-feedback law obtained from GPOPS-II solutions. Across an extensive simulation campaign comprising more than 18,000 manoeuvres, the proposed controllers achieved manoeuvre times within only a few percent of the corresponding time-optimal reference solutions. Moreover, the best-performing architectures outperformed both the Reinforcement Learning and the Nonlinear Model Predictive Control baselines in terms of time-optimality. The simulation campaign also demonstrated robustness to spacecraft inertia uncertainties. Despite being trained exclusively on nominal conditions, the proposed controllers successfully handled both reduced-mass scenarios and inertia-matrix mismatches, maintaining low performance degradation and exhibiting only a very limited number of manoeuvre failures. These results are particularly noteworthy considering the known limitations of behavioural cloning approaches, including distribution-shift effects arising when the controller encounters states not represented in the training dataset. The characterisation was then extended to navigation errors, considering both white sensor noise and the more demanding case of a constant state-estimation offset. The proposed controllers degrade gracefully in both scenarios: across every error model and magnitude tested, no architecture-activation combination exceeds a 0.5% failure rate, whereas both baselines lose reliability as soon as the navigation error becomes correlated in time. The closed-loop behaviour was also assessed by means of a Lyapunov-based analysis. This does not constitute a formal proof of stability, since the controllers are not synthesised to satisfy $\dot{V} \leq 0$ and the closed-loop dynamics remain outside the scope of a rigorous Lyapunov-based theorem. It does, however, provide quantitative and reproducible evidence that the trained controllers behave consistently with energy dissipation towards the desired equilibrium across the full simulation campaign, and it yields a diagnostic that helps explain the failures recorded with the sinusoidal activation. An additional observation emerging from this study concerns the application dependence of neural-network performance. Although SIREN architectures have demonstrated promising characteristics in several aerospace guidance and control applications [58], they exhibited less reliable closed-loop behaviour in the considered time-optimal spacecraft reorientation problem, leading to a small number of manoeuvre failures despite better training performance. This finding suggests that architectural choices that prove effective in one guidance and control application do not necessarily provide the best closed-loop performance in another. The analysis of Section 5 traced this behaviour to the shape of the learned control surface, on which the sinusoidal basis approximates the discontinuous optimal control with a persistent ringing. The result is therefore specific to the architectures and to the training procedure adopted here: since the mechanism lies in the ringing of the learned surface, different training strategies, or deeper or differently regularised networks, may well recover surfaces that render the required switch cleanly, free of the spurious oscillations, and thus mitigate the effect, which was nevertheless observed consistently across all the SIREN-based models considered in this comparison.

From an implementation perspective, all investigated architectures achieved inference times on the order of milliseconds when executed on commercial off-the-shelf CPU-only embedded platforms. The tested G&CNET controllers thus combine two characteristics that are often difficult to achieve simultaneously in autonomous spacecraft guidance and control: a level of optimality remarkably close to the theoretical minimum-time solutions generated by GPOPS-II, and a computational burden reduced, once the training is performed offline, to the evaluation of a lightweight neural network, with no online optimisation involved. This combination makes them attractive candidates for future autonomous spacecraft applications.

Overall, the comparison carried out in this work allows a balanced consideration of the respective merits of the approaches. The G&CNETs deliver the closest reproduction of the time-optimal behaviour and the strongest robustness to navigation errors, but they inherit the fully saturated command of the expert they clone, and hence its maximal propellant and energy expenditure, and their closed-loop stability can at present be assessed only empirically. The RL agents achieve competitive manoeuvre times with the tightest dispersion and require no expert dataset, but optimality can only be induced indirectly through the reward, at the price of a careful reward shaping and of a training budget of hundreds of millions of environment interactions. The NMPC, in turn, is the only method with genuinely smooth commands and an appreciably lower energy expenditure, and requires no training at all, but pays for these properties with a manoeuvre time far from the optimum and an inference cost more than an order of magnitude higher than that of the neural controllers. The choice among the three is therefore dictated by the mission priorities: when the manoeuvre time is the binding requirement, the proposed G&CNETs stand out as the most suitable option, whereas when the propellant budget or the actuator wear dominates and the time budget is relaxed, the NMPC becomes attractive.

Future work will focus on the closed-loop stability of the proposed controllers, whose formal and analytical treatment remains an open problem: as discussed in Section 5, the verification tools currently available do not scale to closed loops of this dimensionality and nonlinearity. While several approaches for the verification and certification of neural-network-based control systems have recently been proposed [84,87,89], establishing rigorous and broadly applicable guarantees remains an active research challenge. Investigating the applicability of these methodologies to the controllers proposed in this work therefore represents a natural direction for future research.

CRediT authorship contribution statement

Luca Bigelli: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Federica Paganelli Azza:** Writing – review & editing, Supervision. **Luca Romanelli:** Writing – review & editing, Supervision. **Mattia Varile:** Writing – review & editing, Supervision. **Marcello Romano:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

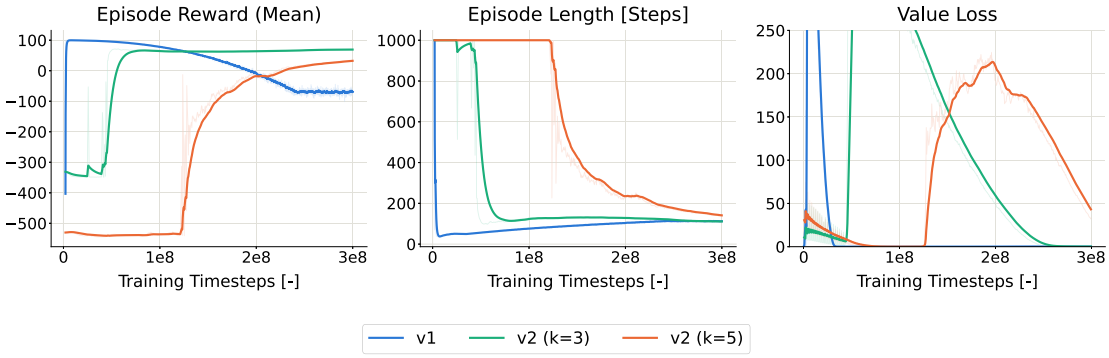


Fig. A.13. Training curves for the three considered RL agents: mean episode reward, mean episode length, and PPO value loss versus training timesteps. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Acknowledgements

For the author Luca Bigelli this publication is part of the project PNRR-NGEU which has received funding from the MUR – DM 117/2023.

Finanziato dall'Unione europea NextGenerationEU | Ministero dell'Università e della Ricerca | Italiadomani

Appendix A. Reinforcement Learning (RL)

As a term of comparison for the presented neural networks, a model-free Reinforcement Learning agent was trained to solve the time-optimal reorientation problem within the Proximal Policy Optimisation (PPO) framework [90]. The agent operates on a continuous 7-dimensional observation space, comprising the attitude error quaternion and the normalised angular velocity vector. The action space is discrete: the agent commands one of three torque levels $\{-T_{max}, 0, T_{max}\}$ per spacecraft axis, yielding $3^3 = 27$ possible combinations per control step. Separate Actor and Critic networks are used, each with three hidden layers of 128 units and Hyperbolic Tangent (*Tanh*) activations. The episodes are simulated with MuJoCo XLA (MJX), the JAX-based rewrite of the MuJoCo physics engine [91], which enables GPU-parallel simulation of thousands of environments. The spacecraft model of Section 2 is controlled at a fixed time step $\Delta t = 0.1$ s. An episode terminates successfully when the angular error drops below 0.5 deg and the angular velocity below 0.5 deg/s (the same nominal thresholds used for simulation throughout this work) and is truncated if the maximum duration of 1000 control steps is reached. No domain randomisation was applied. Curriculum learning was used to progressively expose the agent to harder manoeuvres: the maximum initial slew angle grows over training and saturates at 180° once 80% of the total timesteps have elapsed.

Reward design. Two reward formulations were tested. The first (v1) is a dense quadratic penalty on the distance from the target state,

$$r_t = -\theta_{\mathrm{err},t}^2 - 0.5\,\|\boldsymbol{\omega}_{\mathrm{err},t}\|^2, \tag{A.1}$$

where $\theta_{\mathrm{err},t}$ and $\boldsymbol{\omega}_{\mathrm{err},t}$ are the angular attitude and angular velocity errors at time t; a terminal bonus of +100 is granted on success and a penalty of −50 on timeout. A more time-optimal–oriented reward with a constant per-step penalty $-\Delta t$ was also tested. It however provided no informative gradient: every action yields the same immediate reward regardless of its effect on the attitude error, so the only useful signal is the terminal bonus, which is rarely reached in the early iterations even with a curriculum-learning strategy. As a result, no learning occurred and every episode timed out, so this variant was discarded. To improve this reward function, a potential-based shaping term [71] was added, based on the angle-to-go (v2),

$$r_t = -k\,\Delta t + (\gamma\,\Phi_t - \Phi_{t-1}), \qquad \Phi_t = -\theta_{\mathrm{err},t}, \tag{A.2}$$

with $\gamma = 0.99$ the discount factor used by PPO. The shaping term rewards each step that reduces the attitude error, while the weighted time penalty $-k\,\Delta t$ retains the min-time pressure. Two values of the time weight were tested, $k = 3$ and $k = 5$, the former yielding a better policy.

Training curves. Fig. A.13 reports the mean episode reward, mean episode length, and PPO value loss against training timesteps for the quadratic baseline (v1) and both min-time-shaped runs (v2, $k = 3$ and $k = 5$). As previously mentioned, the curriculum saturates at 80% of the timestep budget; consistently, each curve shows a clear inflection and plateau only in its final segment, once full difficulty has been reached while still leaving enough steps to consolidate. The training budget was initially set to 1×10^8 timesteps but was then raised to 3×10^8, as the value loss of the v2 agents had not yet converged. The extended budget lets v1 and v2 ($k = 3$) reach a stable plateau in both value loss and episode length, but is still insufficient for the v2 agent with $k = 5$: this run shows a higher value loss that has still not plateaued; we therefore kept only the v2 ($k = 3$) variant. Notably, despite the different reward formulations, v1 and v2 ($k = 3$) converge to a very similar final mean episode length.

Testing the two trained agents on the full set of 18432 manoeuvres revealed different but competitive performances, as discussed in Section 4.2, which motivates the inclusion of both models in the comparison.

Appendix B. Nonlinear Model Predictive Control (NMPC)

The Nonlinear Model Predictive Control (NMPC) baseline was implemented with the *acados* solver [92], a fast nonlinear optimal control solver designed for deployment on embedded platforms and widely utilised across various contexts [93–96]. The controller reuses the rigid-body dynamics and kinematics of Eqs. (1) and (2) as the prediction model, formulated on the error state $\mathbf{x}_{\mathrm{err}} = [\mathrm{q}_{\mathrm{err}}\ \boldsymbol{\omega}_{\mathrm{err}}]^\top$. A standard quadratic tracking cost penalises the error state and the control input over a prediction horizon T_p, discretised into N shooting intervals of duration $T_s = T_p/N$:

$$\min_{\mathbf{u}(\cdot)} \sum_{k=0}^{N-1} \left(\|\mathbf{x}_{\mathrm{err},k}\|_{\mathbf{Q}}^2 + \|\mathbf{u}_k\|_{\mathbf{R}}^2 \right) + \|\mathbf{x}_{\mathrm{err},N}\|_{\mathbf{Q}}^2, \tag{B.1}$$

subject to the discretised error-state dynamics and to the per-axis torque box constraints defined in Section 2, with $\mathbf{Q} = \mathrm{diag}(w_q, w_q, w_q, w_q, w_\omega, w_\omega, w_\omega)$ and $\mathbf{R} = \mathrm{diag}(w_c, w_c, w_c)$. The cost weights are set to $w_q = 10$, $w_\omega = 5$, and $w_c = 10$. The prediction horizon is set to $T_p = 1.0$ s and the sampling time to $T_s = 0.1$ s, thus leading to $N = 10$ shooting intervals.

Parameter tuning. The *cost weights* were selected by minimising the mean manoeuvre-time increase relative to the GPOPS-II optimum over a representative subset of manoeuvres spanning the full range of rotation angles, subject to clean convergence. The dominant parameter is the ratio between the angular-velocity and the attitude weight, w_ω/w_q: a value of about 0.5 (here $w_q = 10$, $w_\omega = 5$) yields the fastest, cleanly-convergent response. Increasing w_ω towards w_q makes the controller more conservative, and hence slower, whereas lowering it too far reduces the effective damping and induces overshoot that lengthens the manoeuvre; the control weight w_c proved comparatively insensitive over a wide range. As for the *prediction horizon*, $T_p = 1.0\,\mathrm{s}$ proved to be a sweet spot: shorter horizons are too myopic and converge slowly, whereas longer ones increase the solve time without improving the manoeuvre time. Finally, halving the *sampling time* to $T_s = 0.05\,\mathrm{s}$ (to match the GCNET controllers' rate) led to a slower manoeuvre time, so $T_s = 0.1\,\mathrm{s}$ was retained.

References

[1] J.E. Cochran, Jr., B.K. Colburn, N.O. Speakman, Adaptive spacecraft attitude control utilizing eigenaxis rotations, in: 13th Aerospace Sciences Meeting, 1975, http://dx.doi.org/10.2514/6.1975-158.

[2] P. van den Bosch, W. Jongkind, A. van Swieten, Adaptive attitude control for large-angle slew manoeuvres, Automatica 22 (2) (1986) 209–215, http://dx.doi.org/10.1016/0005-1098(86)90082-8.

[3] L. D'Amario, G. Stubbs, A new single-rotation-axis autopilot for rapid spacecraft attitude maneuvers, J. Guid. Control 2 (4) (1979) 339–346, http://dx.doi.org/10.2514/3.55885.

[4] B. Wie, H. Weiss, A. Arapostathis, Quarternion feedback regulator for spacecraft eigenaxis rotations, J. Guid. Control Dyn. 12 (3) (1989) 375–380, http://dx.doi.org/10.2514/3.20418.

[5] W.H. Steyn, Near-minimum-time eigenaxis rotation maneuvers using reaction wheels, J. Guid. Control Dyn. 18 (5) (1995) 1184–1189, http://dx.doi.org/10.2514/3.21523.

[6] B. Wie, J. Lu, Feedback control logic for spacecraft eigenaxis rotations under slew rate and control constraints, J. Guid. Control Dyn. 18 (6) (1995) 1372–1379, http://dx.doi.org/10.2514/3.21555.

[7] K.D. Bilimoria, B. Wie, Time-optimal three-axis reorientation of a rigid spacecraft, J. Guid. Control Dyn. 16 (3) (1993) 446–452, http://dx.doi.org/10.2514/3.21030.

[8] A. Fleming, P. Sekhavat, I.M. Ross, Minimum-time reorientation of a rigid body, J. Guid. Control Dyn. 33 (1) (2010) 160–170, http://dx.doi.org/10.2514/1.43549.

[9] X. Bai, J.L. Junkins, New results for time-optimal three-axis reorientation of a rigid spacecraft, J. Guid. Control Dyn. 32 (4) (2009) 1071–1076, http://dx.doi.org/10.2514/1.43097.

[10] H. Shen, P. Tsiotras, Time-optimal control of axisymmetric rigid spacecraft using two controls, J. Guid. Control Dyn. 22 (5) (1999) 682–694, http://dx.doi.org/10.2514/2.4436.

[11] R. Proulx, I. Ross, Time-optimal reorientation of asymmetric rigid bodies, Adv. Astronaut. Sci. 109 (2002) 1207–1227.

[12] G. Elnagar, M. Kazemi, M. Razzaghi, The pseudospectral Legendre method for discretizing optimal control problems, IEEE Trans. Autom. Control 40 (10) (1995) 1793–1796, http://dx.doi.org/10.1109/9.467672.

[13] F. Fahroo, I.M. Ross, Costate estimation by a Legendre pseudospectral method, J. Guid. Control Dyn. 24 (2) (2001) 270–277, http://dx.doi.org/10.2514/2.4709.

[14] I.M. Ross, Enhancements to the DIDO optimal control toolbox, 2020, arXiv:2004.13112, URL https://arxiv.org/abs/2004.13112.

[15] I.M. Ross, P. Sekhavat, A. Fleming, Q. Gong, Optimal feedback control: Foundations, examples, and experimental results for a new approach, J. Guid. Control Dyn. 31 (2) (2008) 307–321, http://dx.doi.org/10.2514/1.29532.

[16] A. Fleming, P. Sekhavat, I. Ross, Minimum-time reorientation of an asymmetric rigid body, in: AIAA Guidance, Navigation and Control Conference and Exhibit, 2008-08, http://dx.doi.org/10.2514/6.2008-7012.

[17] B.M. Yutko, R.G. Melton, Optimizing spacecraft reorientation maneuvers using a pseudospectral method, J. Aerosp. Eng. Sci. Appl. 2 (1) (2010) 1–14.

[18] R.G. Melton, Hybrid methods for determining time-optimal, constrained spacecraft reorientation maneuvers, Acta Astronaut. 94 (1) (2014) 294–301, http://dx.doi.org/10.1016/j.actaastro.2013.05.007.

[19] G. Boyarko, O. Yakimenko, M. Romano, Formulation and analysis of matching points of interest in two-spacecraft for optimal rendezvous, in: AIAA Guidance, Navigation, and Control Conference, 2009-08, http://dx.doi.org/10.2514/6.2009-5669.

[20] G. Basset, Y. Xu, O.A. Yakimenko, Computing short-time aircraft maneuvers using direct methods, J. Comput. Syst. Sci. Int. 49 (3) (2010) 481–513, http://dx.doi.org/10.1134/S1064230710030159.

[21] M. Karpenko, S. Bhatt, N. Bedrossian, I.M. Ross, Flight implementation of shortest-time maneuvers for imaging satellites, J. Guid. Control Dyn. 37 (4) (2014) 1069–1079, http://dx.doi.org/10.2514/1.62867.

[22] M. Karpenko, S. Bhatt, N. Bedrossian, A. Fleming, I.M. Ross, First flight results on time-optimal spacecraft slews, J. Guid. Control Dyn. 35 (2) (2012) 367–376, http://dx.doi.org/10.2514/1.54937.

[23] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948, http://dx.doi.org/10.1109/ICNN.1995.488968.

[24] H. Chen, Y. Zhu, K. Hu, Adaptive bacterial foraging optimization, Abstr. Appl. Anal. 2011 (1) (2011) 108269, http://dx.doi.org/10.1155/2011/108269.

[25] M. Pontani, R. Melton, Heuristic optimization of satellite reorientation maneuvers, in: AIAA/AAS Astrodynamics Specialist Conference, 2016, http://dx.doi.org/10.2514/6.2016-5581.

[26] D. Spiller, L. Ansalone, F. Curti, Particle swarm optimization for time-optimal spacecraft reorientation with keep-out cones, J. Guid. Control Dyn. 39 (2) (2016) 312–325, http://dx.doi.org/10.2514/1.G001228.

[27] R.G. Melton, Differential evolution/particle swarm optimizer for constrained slew maneuvers, Acta Astronaut. 148 (2018) 246–259, http://dx.doi.org/10.1016/j.actaastro.2018.04.045.

[28] R. Melton, Constrained time-optimal slewing maneuvers for rigid spacecraft, Adv. Astronaut. Sci. 135 (2010) 107–126.

[29] R.G. Melton, Numerical analysis of constrained, time-optimal satellite reorientation, Math. Probl. Eng. 2012 (1) (2012) 769376, http://dx.doi.org/10.1155/2012/769376.

[30] J. Li, X.-n. Xi, Time-optimal reorientation of the rigid spacecraft using a pseudospectral method integrated homotopic approach, Optim. Control. Appl. Methods 36 (6) (2015) 889–918, http://dx.doi.org/10.1002/oca.2145.

[31] J. Li, Time-optimal three-axis reorientation of asymmetric rigid spacecraft via homotopic approach, Adv. Space Res. 57 (10) (2016) 2204–2217, http://dx.doi.org/10.1016/j.asr.2016.02.016.

[32] C. Louembet, F. Cazaurang, A. Zolghadri, C. Charbonnel, C. Pittet, Design of algorithms for satellite slew manoeuver by flatness and collocation, in: 2007 American Control Conference, 2007, pp. 3168–3173, http://dx.doi.org/10.1109/ACC.2007.4282459.

[33] O. Yakimenko, Optimization of holonomic attitude dynamics using IDVD method, in: 2010 3rd International Symposium on Systems and Control in Aeronautics and Astronautics, 2010, pp. 1496–1499, http://dx.doi.org/10.1109/ISSCAA.2010.5632357.

[34] G.A. Boyarko, M. Romano, O.A. Yakimenko, Time-optimal reorientation of a spacecraft using an inverse dynamics optimization method, J. Guid. Control Dyn. 34 (4) (2011) 1197–1208, http://dx.doi.org/10.2514/1.49449.

[35] J. Ventura, M. Romano, U. Walter, Performance evaluation of the inverse dynamics method for optimal spacecraft reorientation, Acta Astronaut. 110 (2015) 266–278, http://dx.doi.org/10.1016/j.actaastro.2014.11.041, Dynamics and Control of Space Systems.

[36] D. Izzo, M. Märtens, B. Pan, A survey on artificial intelligence trends in spacecraft guidance dynamics and control, Astrodynamics 3 (4) (2019-12) 287–299, http://dx.doi.org/10.1007/s42064-018-0053-6.

[37] M. Shirobokov, S. Trofimov, M. Ovchinnikov, Survey of machine learning techniques in spacecraft control design, Acta Astronaut. 186 (2021) 87–97, http://dx.doi.org/10.1016/j.actaastro.2021.05.018.

[38] S. Silvestrini, M. Lavagna, Deep learning and artificial neural networks for spacecraft dynamics, navigation and control, Drones 6 (10) (2022) http://dx.doi.org/10.3390/drones6100270.

[39] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, C.L.P. Chen, Review of advanced guidance and control algorithms for space/aerospace vehicles, Prog. Aerosp. Sci. 122 (2021) 100696, http://dx.doi.org/10.1016/j.paerosci.2021.100696.

[40] D.A. Pomerleau, ALVINN: an autonomous land vehicle in a neural network, in: Proceedings of the 2nd International Conference on Neural Information Processing Systems, NIPS '88, MIT Press, 1988, pp. 305–313.

[41] D. Michie, in: L.S. Sterling (Ed.), Intelligent Systems: Concepts and Applications, Springer, 1993.

[42] I. Bratko, T. Urbančič, C. Sammut, Behavioural cloning: Phenomena, results and problems, IFAC Proc. Vol. 28 (21) (1995) 143–149, http://dx.doi.org/10.1016/S1474-6670(17)46716-4, 5th IFAC Symposium on Automated Systems Based on Human Skill (Joint Design of Technology and Organisation), Berlin, Germany, 26-28 September.

[43] W. McDermott, M. Athans, Approximating optimal state feedback using neural networks, in: Proceedings of 1994 33rd IEEE Conference on Decision and Control, vol. 3, 1994, pp. 2466–2471, http://dx.doi.org/10.1109/CDC.1994.411510.

[44] D. Izzo, E. Blazquez, R. Ferede, S. Origer, C. De Wagter, G.C.H.E. de Croon, Optimality principles in spacecraft neural guidance and control, Sci. Robot. 9 (91) (2024) eadi6421, http://dx.doi.org/10.1126/scirobotics.adi6421.

[45] C. Sánchez-Sánchez, D. Izzo, Real-time optimal control via deep neural networks: Study on landing problems, J. Guid. Control Dyn. 41 (5) (2018) 1122–1135, http://dx.doi.org/10.2514/1.G002357.

[46] D. Izzo, E. Öztürk, Real-time guidance for low-thrust transfers using deep neural networks, J. Guid. Control Dyn. 44 (2) (2021) 315–327, http://dx.doi.org/10.2514/1.G005254.

[47] D. Izzo, S. Origer, Neural representation of a time optimal, constant acceleration rendezvous, Acta Astronaut. 204 (2023) 510–517, http://dx.doi.org/10.1016/j.actaastro.2022.08.045.

[48] L. Cheng, Z. Wang, F. Jiang, C. Zhou, Real-time optimal control for spacecraft orbit transfer via multiscale deep neural networks, IEEE Trans. Aerosp. Electron. Syst. 55 (5) (2019) 2436–2450, http://dx.doi.org/10.1109/TAES.2018.2889571.

[49] L. Cheng, Z. Wang, Y. Song, F. Jiang, Real-time optimal control for irregular asteroid landings using deep neural networks, Acta Astronaut. 170 (2020) 66–79, http://dx.doi.org/10.1016/j.actaastro.2019.11.039.

[50] W. Li, Y. Song, L. Cheng, S. Gong, Closed-loop deep neural network optimal control algorithm and error analysis for powered landing under uncertainties, Astrodynamics 7 (2) (2023) 211–228, http://dx.doi.org/10.1007/s42064-022-0153-1.

[51] J.D. Biggs, H. Fournier, Neural-network-based optimal attitude control using four impulsive thrusters, J. Guid. Control Dyn. 43 (2) (2020) 299–309, http://dx.doi.org/10.2514/1.G004226.

[52] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, C.L.P. Chen, Six-DOF spacecraft optimal trajectory planning and real-time attitude control: A deep neural network-based approach, IEEE Trans. Neural Netw. Learn. Syst. 31 (11) (2020) 5005–5013, http://dx.doi.org/10.1109/TNNLS.2019.2955400.

[53] D. Tailor, D. Izzo, Learning the optimal state-feedback via supervised imitation learning, Astrodynamics 3 (4) (2019) 361–374, http://dx.doi.org/10.1007/s42064-019-0054-0.

[54] S. Li, E. Öztürk, C. De Wagter, G.C.H.E. de Croon, D. Izzo, Aggressive online control of a quadrotor via deep network representations of optimality principles, in: 2020 IEEE International Conference on Robotics and Automation, ICRA, 2020, pp. 6282–6287, http://dx.doi.org/10.1109/ICRA40945.2020.9197443.

[55] R. Ferede, G. de Croon, C. De Wagter, D. Izzo, End-to-end neural network based optimal quadcopter control, Robot. Auton. Syst. 172 (2024) 104588, http://dx.doi.org/10.1016/j.robot.2023.104588.

[56] V. Sitzmann, J.N.P. Martel, A.W. Bergman, D.B. Lindell, G. Wetzstein, Implicit neural representations with periodic activation functions, 2020, arXiv:2006.09661.

[57] S. Origer, C.D. Wagter, R. Ferede, G.C.H.E. de Croon, D. Izzo, Guidance & control networks for time-optimal quadcopter flight, 2023, arXiv:2305.02705.

[58] S. Origer, D. Izzo, Guidance and control networks with periodic activation functions, Astrodynamics (2026) http://dx.doi.org/10.1007/s42064-025-0285-1.

[59] G. Tang, W. Sun, K. Hauser, Learning trajectories for real- time optimal control of quadrotors, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2018, pp. 3620–3625, http://dx.doi.org/10.1109/IROS.2018.8593536.

[60] L. Federici, B. Benedikter, A. Zavoli, Deep learning techniques for autonomous spacecraft guidance during proximity operations, J. Spacecr. Rockets 58 (6) (2021) 1774–1785, http://dx.doi.org/10.2514/1.A35076.

[61] H. Holt, S. Origer, D. Izzo, Comparing behavioral cloning and reinforcement learning for spacecraft guidance and control networks, J. Guid. Control Dyn. 49 (7) (2026) 1927–1942, http://dx.doi.org/10.2514/1.G009433.

[62] S. Ross, D. Bagnell, Efficient reductions for imitation learning, in: Y.W. Teh, M. Titterington (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, vol. 9, PMLR, 2010, pp. 661–668.

[63] S. Ross, G.J. Gordon, J.A. Bagnell, A reduction of imitation learning and structured prediction to no-regret online learning, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS, in: Proceedings of Machine Learning Research, vol. 15, 2011, pp. 627–635.

[64] J.G. Elkins, R. Sood, C. Rumpf, Adaptive continuous control of spacecraft attitude using deep reinforcement learning, in: AAS/AIAA Astrodynamics Specialist Conference, in: Advances in the Astronautical Sciences, vol. 179, Univelt, Inc., 2020, pp. AAS 20–475.

[65] J.G. Elkins, R. Sood, C. Rumpf, Autonomous spacecraft attitude control using deep reinforcement learning, in: Proceedings of the 71st International Astronautical Congress, IAC, in: The CyberSpace Edition, International Astronautical Federation (IAF), 2020-10, pp. IAC–20–C1.9.8.

[66] F. Vedant, A. James T., M. West, A. Ghosh, Reinforcement learning for spacecraft attitude control, in: Proceedings of the 70th International Astronautical Congress, IAC, International Astronautical Federation (IAF), 2019-10.

[67] B. Gaudet, R. Linares, R. Furfaro, Adaptive guidance and integrated navigation with reinforcement meta-learning, Acta Astronaut. 169 (2020) 180–190, http://dx.doi.org/10.1016/j.actaastro.2020.01.007.

[68] B. Gaudet, R. Linares, R. Furfaro, Terminal adaptive guidance via reinforcement meta-learning: Applications to autonomous asteroid close-proximity operations, Acta Astronaut. 171 (2020) 1–13, http://dx.doi.org/10.1016/j.actaastro.2020.02.036.

[69] K. Hovell, S. Ulrich, Deep reinforcement learning for spacecraft proximity operations guidance, J. Spacecr. Rockets 58 (2) (2021) 254–264, http://dx.doi.org/10.2514/1.A34838.

[70] R. Ferede, C. De Wagter, D. Izzo, G.C.H.E. de Croon, End-to-end reinforcement learning for time-optimal quadcopter flight, in: 2024 IEEE International Conference on Robotics and Automation, ICRA, 2024, http://dx.doi.org/10.1109/ICRA57147.2024.10611665.

[71] A.Y. Ng, D. Harada, S. Russell, Policy invariance under reward transformations: Theory and application to reward shaping, in: Proceedings of the Sixteenth International Conference on Machine Learning, 1999, pp. 278–287.

[72] H. Leeghim, Y. Choi, H. Bang, Adaptive attitude control of spacecraft using neural networks, Acta Astronaut. 64 (7) (2009) 778–786, http://dx.doi.org/10.1016/j.actaastro.2008.12.004.

[73] H. Dong, X. Zhao, H. Yang, Reinforcement learning-based approximate optimal control for attitude reorientation under state constraints, IEEE Trans. Control Syst. Technol. 29 (4) (2021) 1664–1673, http://dx.doi.org/10.1109/TCST.2020.3007401.

[74] H. Yang, Q. Hu, H. Dong, X. Zhao, ADP-based spacecraft attitude control under actuator misalignment and pointing constraints, IEEE Trans. Ind. Electron. 69 (9) (2022) 9342–9352, http://dx.doi.org/10.1109/TIE.2021.3116571.

[75] T. Nakamura-Zimmerer, Q. Gong, W. Kang, Adaptive deep learning for high-dimensional Hamilton–Jacobi–Bellman equations, SIAM J. Sci. Comput. 43 (2) (2021) A1221–A1247, http://dx.doi.org/10.1137/19M1288802.

[76] A. D'Ambrosio, R. Furfaro, Learning fuel-optimal trajectories for space applications via pontryagin neural networks, Aerospace 11 (3) (2024) 228, http://dx.doi.org/10.3390/aerospace11030228.

[77] K. Wang, Z. Chen, J. Li, Fuel-optimal powered descent guidance for lunar pinpoint landing using neural networks, Adv. Space Res. 74 (10) (2024) 5006–5022, http://dx.doi.org/10.1016/j.asr.2024.07.019.

[78] M.A. Patterson, A.V. Rao, GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming, ACM Trans. Math. Software 41 (1) (2014-10) http://dx.doi.org/10.1145/2558904.

[79] H. Schaub, J.L. Junkins, Analytical mechanics of space systems, in: AIAA Education Series, American Institute of Aeronautics and Astronautics, Inc., 2018, http://dx.doi.org/10.2514/4.105210.

[80] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, arXiv:1412.6980, URL https://arxiv.org/abs/1412.6980.

[81] J.L. Junkins, T.J. D., Optimal spacecraft rotational maneuvers, in: Studies in Astronautics, North Holland, 2012.

[82] K.M. Górski, E. Hivon, A.J. Banday, B.D. Wandelt, F.K. Hansen, M. Reinecke, M. Bartelmann, HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere, Astrophys. J. 622 (2) (2005-04) 759, http://dx.doi.org/10.1086/427976.

[83] H. Li, X. Zhong, B. Hu, H. Zhang, Two-stage learning of stabilizing neural controllers via zubov sampling and iterative domain expansion, in: The Thirty-Ninth Annual Conference on Neural Information Processing Systems, 2025, URL https://openreview.net/forum?id=IT12Radlnq.

[84] K. Wang, R. Armellin, A. Evans, H. Holt, Z. Chen, Learning-based optimal guidance for spacecraft close-proximity operations with certified stability, Acta Astronaut. 244 (2026) 1–19, http://dx.doi.org/10.1016/j.actaastro.2026.02.009.

[85] D. Fragopoulos, M. Innocenti, Stability considerations in quaternion attitude control using discontinuous Lyapunov functions, IEE Proc. - Control. Theory Appl. 151 (2004) 253–258, http://dx.doi.org/10.1049/ip-cta:20040311.

[86] S. Origer, D. Izzo, G. Acciarini, F. Biscani, R. Mastroianni, M. Bannach, H. Holt, Certifying guidance & control networks: Uncertainty propagation to an event manifold, 2024, arXiv:2410.03729, URL https://arxiv.org/abs/2410.03729.

[87] A. Evans, R. Armellin, Sample-free safety assessment of neural network controllers via taylor methods, 2026, arXiv:2602.11332, URL https://arxiv.org/abs/2602.11332.

[88] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, L. Daniel, Efficient neural network robustness certification with general activation functions, in: Advances in Neural Information Processing Systems (NeurIPS), vol. 31, 2018.

[89] D. Izzo, D. Tailor, T. Vasileiou, On the stability analysis of deep neural network representations of an optimal state feedback, IEEE Trans. Aerosp. Electron. Syst. 57 (1) (2021) 145–154, http://dx.doi.org/10.1109/TAES.2020.3010670.

[90] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv:1707.06347, URL https://arxiv.org/abs/1707.06347.

[91] E. Todorov, T. Erez, Y. Tassa, Mujoco: A physics engine for model-based control, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 5026–5033, http://dx.doi.org/10.1109/IROS.2012.6386109.

[92] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, M. Diehl, acados – a modular open-source framework for fast embedded optimal control, Math. Program. Comput. (2021).

[93] N. Rathod, A. Bratta, M. Focchi, M. Zanon, O. Villarreal, C. Semini, A. Bemporad, Model predictive control with environment adaptation for legged locomotion, IEEE Access 9 (2021) 145710–145727, http://dx.doi.org/10.1109/access.2021.3118957.

[94] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, M. Ryll, Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms, IEEE Robot. Autom. Lett. 8 (4) (2023) 2397–2404, http://dx.doi.org/10.1109/LRA.2023.3246839.

[95] V. Wüest, S. Jeger, M. Feroskhan, E. Ajanic, F. Bergonti, D. Floreano, Agile perching maneuvers in birds and morphing-wing drones, Nat. Commun. 15 (1) (2024) 8330, http://dx.doi.org/10.1038/s41467-024-52369-4.

[96] D. Kloeser, T. Schoels, T. Sartor, A. Zanelli, G. Prison, M. Diehl, NMPC for racing using a singularity-free path-parametric model with obstacle avoidance, IFAC-PapersOnLine 53 (2) (2020) 14324–14329, http://dx.doi.org/10.1016/j.ifacol.2020.12.1376, 21st IFAC World Congress.