Transformer-based Neural Predictive Control for Small-Body Proximity Operations

Tommaso Cesarini
Department of Aerospace Science and Technology
Politecnico di Milano
Milan, Italy
tommaso.cesarini@mail.polimi.it

Stefano Silvestrini
Department of Aerospace Science and Technology
Politecnico di Milano
Milan, Italy
stefano.silvestrini@polimi.it

***Abstract*—Model Predictive Control (MPC) is a powerful control strategy, but its performance is highly dependent on the accuracy of the predictive model and can be limited by significant computational complexity. These limitations become critical in spacecraft proximity operations around small bodies, where the environment is complex and uncertain. To address these issues, this paper proposes a hybrid control framework that integrates a Transformer-based neural network with MPC. The neural network is employed as a surrogate dynamical model, while MPC ensures stability and optimality in control generation through the explicit handling of constraints and uncertainties. The Transformer architecture demonstrates sufficient expressive capability to accurately capture spacecraft dynamics in the vicinity of the reference trajectory, while its parallel computation capabilities enable a significant reduction in computational time compared to traditional nonlinear MPC (NMPC) approaches. Moreover, the proposed framework supports online learning of the system dynamics, allowing real-time adaptation to environmental perturbations and modeling errors. Extensive numerical simulations and Processor-in-the-Loop (PIL) experiments are conducted to validate the proposed approach, demonstrating its effectiveness and potential for enabling autonomous spacecraft operations in close proximity to small bodies.**

***Index Terms*—Model Predictive Control, Deep Learning, Transformer, Online Learning**

I. Introduction

Small-body proximity operations represent some of the most challenging scenarios in spacecraft guidance and control. These missions are conducted in highly uncertain and complex environments, characterized by irregular gravitational fields, poorly known physical parameters, and significant external disturbances. In this context, Model Predictive Control (MPC) represents a suitable approach for trajectory tracking, as it enables the handling of nonlinear dynamics, state and control constraints, and real-time adaptation of control inputs through the repeated solution of an optimal control problem [1]. However, MPC performance strongly depends on the accuracy of the underlying predictive model and is sensitive to unmodeled dynamics and parameter uncertainties. Moreover, the computational burden associated with solving an optimization problem at each control step limits its applicability on resource-constrained platforms such as CubeSats. To overcome these limitations, this work explores the integration of deep learning within an MPC-based control scheme. Deep neural networks offer an efficient representation of the highly nonlinear and uncertain dynamics typical of small-body environments, while online learning capabilities allow continuous adaptation as new data become available [2]. More recently, increasing attention has been devoted to Transformer-based architectures, which represent a paradigm shift compared to traditional sequential models such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks. Unlike these approaches, Transformers eliminate recurrence by leveraging self-attention mechanisms. This allows all elements of a sequence to interact simultaneously, enabling the model to capture long-range dependencies more effectively while significantly improving computational parallelism. In the context of MPC, recent studies have explored the use of Transformers as surrogate models for system dynamics, demonstrating advantages in multi-step prediction accuracy and computational efficiency compared to LSTM-based approaches [3], [4]. Furthermore, in the field of spacecraft guidance and control, several works have recently investigated Transformer architectures for trajectory generation, showing promising results [5], [6]. This paper positions itself within this research direction, with a specific focus on spacecraft control, by investigating the integration of a Transformer-based neural network within an MPC framework for small-body proximity operations. The objective is to assess whether the representational power and parallel structure of Transformers can match the accuracy of traditional nonlinear MPC (NMPC) in such environments while improving computational efficiency, and to evaluate the associated advantages and limitations.

II. Methodology

A. Dynamic models

To test and validate the proposed control algorithm, a representative model of the asteroid environment is required. A polyhedral gravity approach is therefore adopted, enabling accurate computation of the gravitational potential of irregularly shaped small bodies [8]. The analysis focuses on asteroid 433 Eros, a near-Earth asteroid extensively characterized by the *NEAR Shoemaker* mission, whose elongated and irregular shape results in strongly nonlinear gravitational dynamics [9].

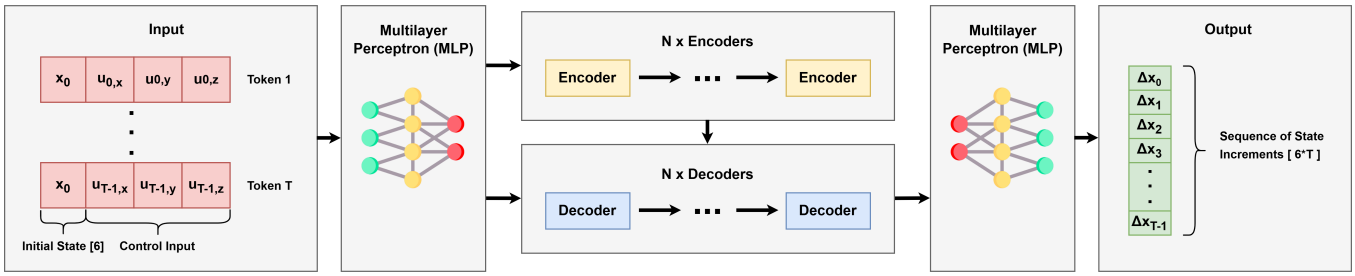


Fig. 1: Transformer-based neural network architecture.

The spacecraft translational dynamics [10], expressed in the asteroid body-fixed frame $\mathcal{T}_b = \text{span}\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$, is given by:

$$\ddot{\mathbf{r}} = \mathbf{a}_G(\mathbf{r}) - 2\,\boldsymbol{\omega} \times \dot{\mathbf{r}} - \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})\,, \tag{1}$$

where $\boldsymbol{\omega}$ denotes the constant asteroid angular velocity vector and $\mathbf{a}_G(\mathbf{r})$ is the gravitational acceleration computed via the polyhedral gravity model. In this work, only gravitational effects are considered, while additional perturbations are left to future developments. Two dynamical representations of asteroid Eros are considered:

- **High-Fidelity (HF) Model**: The HF model is used to simulate the true spacecraft dynamics. It employs the real Eros asteroid physical parameters together with a detailed polyhedral shape mesh composed of 498 triangular facets, derived from *NEAR Shoemaker* mission data [11].
- **Low-Fidelity (LF) Model**: The LF model represents a simplified, pre-mission description of the asteroid environment and is adopted during the neural network training phase to emulate imperfect prior knowledge. Its parameters are obtained by perturbing the true values according to Table I. The spin-axis orientation is defined by two angles (tilt and azimuth), which describe the inclination of the rotation axis with respect to the asteroid principal axis of inertia $\mathbf{b}_3$.

TABLE I: LF model asteroid parameters

Parameter	LF Model Variation
Density ρ	$+10\% \rightarrow \rho_{\text{LF}} = 2937$ kg/m^3
Angular velocity Ω	$+1\% \rightarrow \Omega_{\text{LF}} = 3.35 \times 10^{-4}$ rad/s
Spin axis orientation	tilt $= 3°$, azimuth $= 0°$
Asteroid mesh facets	98

B. Neural network architecture

The proposed architecture for onboard trajectory prediction builds upon [12], with modifications in the input representation and activation functions. The network predicts the spacecraft trajectory over a finite horizon by mapping the initial state and a sequence of control inputs to the corresponding future states. The input is represented as a sequence of tokens of length T, corresponding to the number of discrete time instants within the prediction horizon. Each token is constructed by concatenating the initial state $\mathbf{x}_0 \in \mathbb{R}^6$ with the control input $\mathbf{u}_i \in \mathbb{R}^3$ at time step i, representing an impulsive velocity increment:

$$\mathbf{z}_i = [\,\mathbf{x}_0 \,||\, \mathbf{u}_i\,], \quad i = 0, \ldots, T-1. \tag{2}$$

The network outputs state increments $\Delta\mathbf{x}_i$, from which the full trajectory is reconstructed recursively as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k, \quad k = 0, \ldots, T-1. \tag{3}$$

Unlike conventional autoregressive encoder–decoder Transformers, the proposed architecture employs a non-autoregressive decoder, enabling simultaneous prediction across all time steps. As illustrated in Fig. 1, the decoder receives a latent embedding generated by an initial multilayer perceptron (MLP), together with the memory matrix produced by the encoder. Both the encoder and decoder follow the standard Transformer architecture introduced in [7]. To ensure stable training and well-behaved gradients, the Gaussian Error Linear Unit (GELU) activation function is employed throughout the network, avoiding non-smooth activations such as ReLU [13]. Positional encodings are added to the token embeddings to preserve temporal ordering, following [7]. Table II reports the neural network hyperparameters used to obtain the results presented in this work.

TABLE II: Neural network hyperparameters

NN Hyperparameters	
Input Dimension	$[1, 15, 6]$
Embedding Size	512
Transformer Blocks (N)	2
Attention Heads	8
MLP Hidden Size	2048
Output Dimension	$[1, 15, 6]$

C. Neural network loss function

The adopted loss function enforces both local one-step prediction accuracy and global consistency of the reconstructed trajectory over a prediction horizon of length L. The system state is defined as $\mathbf{x} = [\mathbf{p}, \mathbf{v}]^\top \in \mathbb{R}^6$, where $\mathbf{p} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$ denote position and velocity, respectively. At each time step $i \in \{1, \ldots, L\}$, the network predicts a state increment $\Delta\hat{\mathbf{x}}_i$ approximating the true variation $\Delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$. Local

prediction accuracy is enforced through a weighted mean squared error on the predicted state increments:

$$\mathcal{L}_{\Delta} = \sum_{i=1}^{L} w_i \left(\lambda_p \|\Delta\hat{\mathbf{p}}_i - \Delta\mathbf{p}_i\|_2^2 + \lambda_v \|\Delta\hat{\mathbf{v}}_i - \Delta\mathbf{v}_i\|_2^2 \right) \quad (4)$$

A linearly decaying weighting scheme is adopted to emphasize early prediction steps. To limit long-term drift, the predicted increments are integrated to reconstruct the trajectory, which is then used to define the global loss component:

$$\hat{\mathbf{x}}_i = \mathbf{x}_0 + \sum_{j=1}^{i} \Delta\hat{\mathbf{x}}_j, \quad (5)$$

$$\mathcal{L}_{\text{traj}} = \frac{1}{L} \sum_{i=1}^{L} \left(\lambda_p \|\hat{\mathbf{p}}_i - \mathbf{p}_i\|_2^2 + \lambda_v \|\hat{\mathbf{v}}_i - \mathbf{v}_i\|_2^2 \right). \quad (6)$$

The overall data-driven objective is finally defined as

$$\mathcal{L}_{\text{data}} = \alpha\, \mathcal{L}_{\Delta} + \beta\, \mathcal{L}_{\text{traj}}, \qquad \alpha, \beta > 0, \quad (7)$$

where α and β balance short-term accuracy and long-term consistency.

D. Dataset generation

Since the proposed framework relies on supervised learning, a representative training dataset is required. To this end, a synthetic dataset of fixed-length spacecraft trajectory segments (windows) around the irregular small body is generated, starting from the reference trajectory to be tracked. A set of states is first extracted from the nominal path and then perturbed to define the initial conditions of the trajectory windows. These initial states are subsequently propagated by applying small randomized control inputs. Initial perturbations are sampled from bounded uniform distributions to ensure that all trajectories remain close to, but not coincident with, the reference orbit, while random impulsive velocity increments emulate realistic control actions. To guarantee physical consistency, a rejection-sampling strategy is employed: trajectory samples are discarded if they deviate excessively from the reference path, exceeding a prescribed displacement threshold. This ensures that the dataset contains only dynamically feasible trajectories representative of local proximity operations around the nominal orbit. All valid trajectories are expressed in the asteroid body-fixed reference frame and normalized into non-dimensional units to improve numerical stability during training. Each dataset sample therefore consists of:

- a spacecraft trajectory window, including position and velocity vectors over the selected time horizon;
- the corresponding control input sequence applied over the same interval.

The training dataset is generated using the LF dynamical model, reflecting the limited and uncertain knowledge available during the pre-mission phase.

E. MPC implementation

The nonlinear MPC controller embedding the proposed neural network (NN) is implemented following a standard implicit MPC formulation [1]. An important advantage of integrating the NN within the MPC framework is that the resulting objective function is fully differentiable with respect to the control sequence. This property enables the use of automatic differentiation to compute exact gradients, which are directly supplied to the optimizer. Regarding constraint handling, the nonlinear MPC problem enforces only bound constraints on the optimization variables, reflecting actuator limitations. The resulting nonlinear programming problem is solved using IPOPT (Interior Point OPTimizer) [14], a widely adopted open-source solver for nonlinear optimization based on a primal–dual interior-point method. To provide a meaningful benchmark, a linear time-invariant (LTI) MPC and a conventional nonlinear MPC (NMPC) controller are also implemented for comparison. In the latter case, the predictive model is directly based on the nonlinear dynamics described in Eq. 1, which are numerically integrated within the optimization routine. The physical parameters and the gravitational acceleration model correspond to those of the LF model introduced in Section II-A. Regarding gradient computation, in the standard NMPC the gradients are also evaluated using the PyTorch automatic differentiation framework. Table III reports the MPC settings used for the two testing scenarios presented in Section III-A.

TABLE III: MPC settings

MPC Parameters		
	Scenario 1	**Scenario 2**
Sampling Time T_s	5 min	1 min
Prediction Horizon N_p	80 min	16 min
Control Horizon N_c	80 min	16 min
Weight Matrices		
State Error Weight (Q_{pos} / Q_{vel})	$0.1/0.15$	
Control Weight R	5×10^{-7}	
Terminal State Weight (Z_{pos} / Z_{vel})	$0.1/0.15$	

F. Online learning

The proposed control framework incorporates an online learning mechanism that enables the neural network predictor to adapt to discrepancies between the modelled dynamics and the actual system behavior observed during execution. To enable this adaptation, a dataset is constructed online during closed-loop operation. The applied control inputs, together with the measured spacecraft states, are recorded to construct short trajectory segments over a fixed time horizon. These segments are used as supervised training data for the online refinement of the neural network model. The learning procedure follows the same formulation adopted during the offline phase; however, the loss function is evaluated based on the prediction error between the actual trajectory and the one predicted by the network over the same horizon. To ensure closed-loop stability, online learning is activated only after an initial transient phase, allowing a sufficient

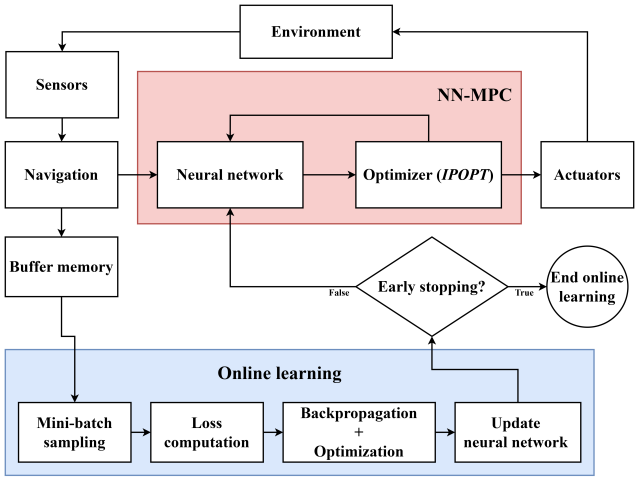


Fig. 2: Control algorithm and online learning schematics.

amount of data to be collected and preventing premature model updates. The collected data are stored in a finite buffer that retains only the most recent trajectory windows while discarding older samples, thereby bounding memory usage and supporting onboard implementation. Once enabled, model adaptation is performed periodically at fixed time intervals rather than continuously, further reducing the computational burden. At each update, only the parameters of the final MLP layer are refined through a limited number of gradient-based optimization steps using mini-batches sampled from the buffer. This design choice significantly reduces computational complexity and is supported by empirical evidence showing no substantial performance improvement when updating the entire network online.

Moreover, to further guide the online learning process toward physically consistent solutions, this work investigates the inclusion of a physics-informed term within the online loss function. By embedding prior knowledge of the system dynamics into the learning objective, the network is encouraged to generate predictions that remain consistent with the underlying laws of motion [15]. To this end, the spacecraft translational dynamics in the rotating body-fixed frame can be rewritten as:

$$\ddot{\mathbf{r}}_{\mathrm{fd}} - \mathbf{a}_G(\mathbf{r}) + 2\,\boldsymbol{\omega} \times \dot{\mathbf{r}} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) = \mathbf{d}_{\mathrm{res}}, \qquad (8)$$

where the residual vector $\mathbf{d}_{\mathrm{res}}$ quantifies the violation of the translational dynamics and, therefore, in the absence of modelling errors, should ideally vanish. The physics-informed term is constructed by computing these residuals and penalizing their magnitude within the online learning loss function. In this formulation, the position $\mathbf{r}$ and velocity $\dot{\mathbf{r}}$ are obtained from the network state predictions. The term $\ddot{\mathbf{r}}_{\mathrm{fd}}$ denotes a finite-difference approximation of the acceleration in the asteroid body frame, computed from the predicted velocities, while $\mathbf{a}_G(\mathbf{r})$ represents the gravitational acceleration evaluated via a truncated spherical harmonics expansion [8], [16]. It is important to note that key dynamical parameters, such as the asteroid density ρ, angular velocity $\boldsymbol{\omega}$, spin-axis orientation, and the coefficients defining the gravitational potential, are initially estimated on ground with limited accuracy. Therefore, prior to enforcing the physics-informed constraint on the network predictions, these parameters are refined through online estimation. To this end, the dynamical parameters are treated as trainable variables and optimized by minimizing the same physics-based residuals in (8), computed using state measurements stored in the buffer memory rather than network predictions.

G. Experimental set-up (PIL)

Processor-in-the-Loop (PIL) simulations are performed to assess the performance of the proposed algorithm on hardware representative of onboard computational constraints [17]. In this context, two computational platforms are considered and their performance is compared, with particular focus on the degradation in execution time when transitioning from a general-purpose computing platform to the more representative onboard hardware.

- Laptop:
  - Intel Core i5-12500H (12th Generation, 2.50 GHz),
  - 16 GB RAM,
  - NVIDIA GeForce RTX 3050 Laptop GPU (4 GB).
- NVIDIA Jetson Orin Nano:
  - 6-core ARM Cortex-A78AE CPU,
  - 8 GB LPDDR5 memory,
  - NVIDIA Ampere GPU (1024 CUDA cores, 32 Tensor Cores).

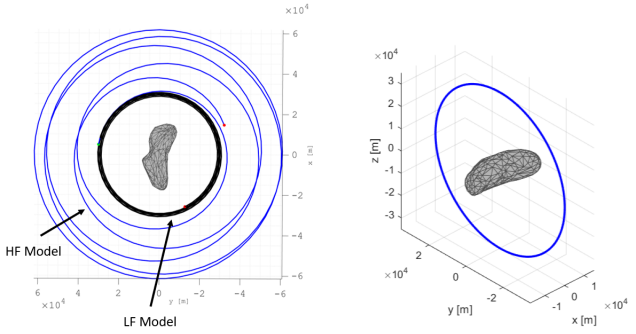
Although not space-qualified, the NVIDIA Jetson Orin Nano exhibits computational architecture comparable to emerging onboard processors for AI-driven autonomy. NVIDIA Jetson–class hardware is already being adopted in current and planned space missions, suggesting that similar technologies may play a key role in future autonomous applications, including proximity operations around small bodies. Specifically, during PIL experiments, the Jetson hosts the complete onboard control algorithm implemented in Python, while the laptop executes the high-fidelity environment model. The latter propagates the spacecraft dynamics, applies the control inputs computed by the Jetson, and returns the resulting system state. On the onboard platform, the optimization routine is executed on the CPU, whereas the evaluation of the cost function and its gradients is accelerated on the GPU.

III. Results

A. Testing scenarios

For the testing of this algorithm, two different scenarios were considered, as described below.

- **Scenario 1:** a quasi-periodic orbit around the asteroid, computed using the LF, is considered as the reference trajectory. Due to the discrepancies between the LF and HF models, when the spacecraft is deployed in the real environment (HF), the reference orbit is not maintained without active control. A sampling time of 5 minutes is



(a) Scenario 1 (b) Scenario 2

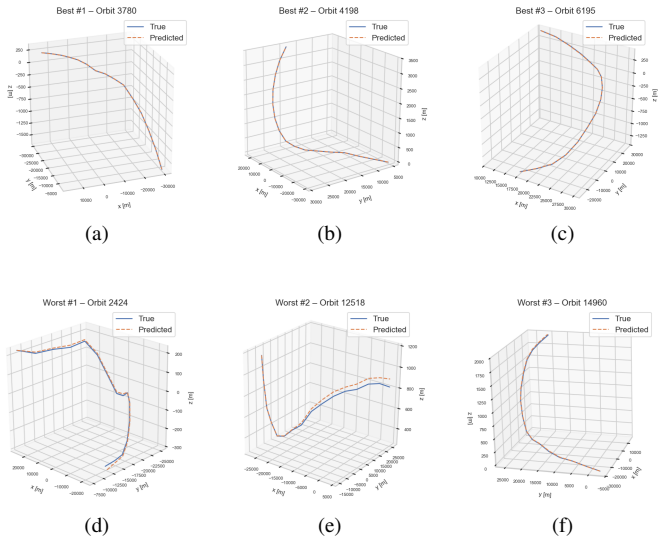
Fig. 3: Reference orbits considered in the MPC simulations.

adopted, together with a prediction horizon of 16 time steps, corresponding to approximately 1.3 hours.

- **Scenario 2:** a circular orbit around the asteroid, lying on the yz plane. In this case a prediction horizon with a lower length has been considered. A sampling time of 1 minute and a prediction horizon of 16 time steps are adopted, corresponding to a total horizon length of 16 minutes.

B. Net training

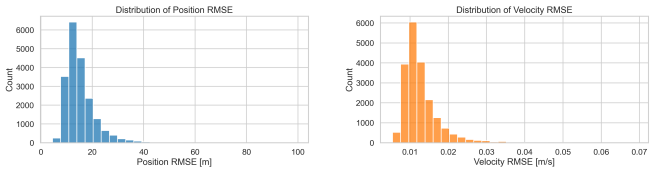
This section reports the results obtained after training the neural network, with specific reference to the model employed in Scenario 1. Two distinct datasets are used: a training dataset consisting of 100,000 trajectories and a validation dataset composed of 20,000 trajectories. Performance is assessed in terms of the Root Mean Square Error (RMSE) of the predicted position and velocity, computed with respect to the corresponding ground-truth trajectories. The RMSE provides a quantitative measure of the average prediction accuracy over the considered prediction horizon. For this scenario, the mean position RMSE is 15.350 m, while the mean velocity RMSE



(a) (b) (c)

(d) (e) (f)

Fig. 4: Best (top row) and worst (bottom row) trajectory approximations produced by the neural network (scenario 1).



(a) Position RMSE distribution (b) Velocity RMSE distribution

Fig. 5: Distribution of position and velocity RMSE over the test dataset.

is 0.0128 m/s. Fig. 5 illustrates the distributions of the position and velocity RMSE. As shown, the majority of the trajectories exhibit low prediction errors, while large errors are confined to a limited number of cases in the distribution tails. This indicates that high prediction errors occur only in a small subset of test scenarios and do not significantly affect the overall performance of the model.

C. MPC simulations

This section presents the results in terms of orbit tracking accuracy and fuel efficiency obtained by testing the proposed control framework in two operational scenarios. The results refer to the neural network model trained offline, in order to evaluate tracking performance when the controller relies on a model affected by ground-based modeling inaccuracies. The performance metrics adopted in the following analyses, Absolute Position Error (APE) and Mean Position Error (MPE), are defined as follows:

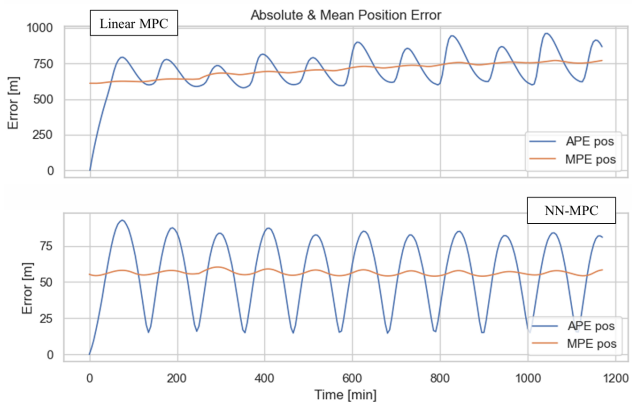
$$\mathrm{APE}_p(t_i) = \|\mathbf{r}_{\mathrm{ref}}(t_i) - \mathbf{r}(t_i)\|_2 \tag{9}$$

$$\mathrm{MPE}_p(t_i) = \frac{1}{N_W} \sum_{j \in W_i} \mathrm{APE}_p(t_j) \tag{10}$$

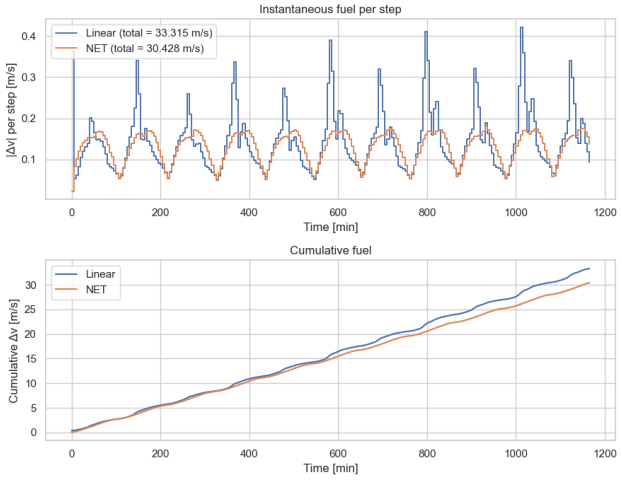
The moving horizon W_i is defined as a centered sliding temporal window around time t_i, spanning $N_W = 100$ discrete time steps. The integration of the neural network within the MPC framework leads to a significant improvement in tracking accuracy compared to the linear MPC case, as shown in Fig. 6a. This result is consistent with the ability of the neural network to accurately approximate the nonlinear dynamics of the asteroid. In addition, the proposed approach achieves a substantial reduction in fuel consumption, as illustrated in Fig. 6b. A comparison in terms of computational time and control performance between the neural network-based MPC and a standard NMPC is also performed. The results for Scenario 1, considering a zero tilt angle, are reported in Table IV.

TABLE IV: Comparison of computational performance and tracking accuracy

Controller	Average Time (Optimization) [s]	MPE [m]
NN-MPC	0.826	13.502
Standard NMPC	28.498	12.750



(a) Tracking error comparison (scenario 1).



(b) Fuel consumption comparison (scenario 1)

Fig. 6: Tracking performance and fuel consumption comparison for scenario 1.

The minimal difference observed in the MPE confirms the capability of the neural network to accurately capture the system dynamics, while providing a significant improvement in computational efficiency. A sensitivity analysis is conducted to assess the impact of errors in the ground-based estimation of key dynamical parameters, namely the asteroid density ρ, angular velocity $\boldsymbol{\omega}$, and spin-axis orientation. The results indicate that the dominant source of performance degradation is the angular mismatch between the assumed and true spin-axis orientation. When this uncertainty is removed, the controller achieves, in Scenario 1, an MPE of 13.5 m. Finally, robustness is assessed through a Monte Carlo analysis comprising 1500 randomized scenarios.

TABLE V: Parameters variations in the Monte Carlo analysis

Parameter	**Range**
Density ρ	$\pm 30\% \rightarrow [2055.9,\ 3818.1]$ kg/m^3
Angular velocity Ω	$\pm 5\% \rightarrow [3.18,\ 3.51] \times 10^{-4}$ rad/s
Spin-axis tilt	$[0,\ 6]$ deg
Spin-axis azimuth	$[0,\ 360]$ deg

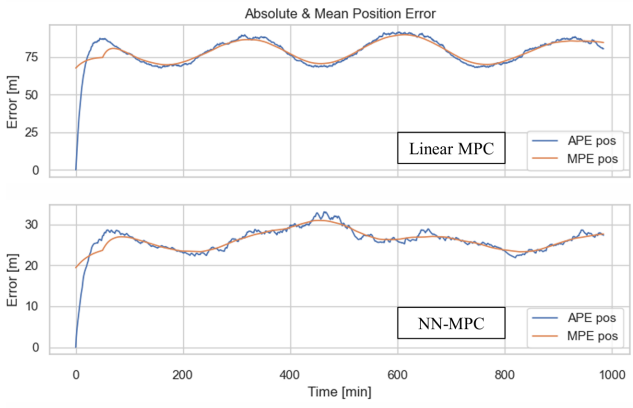


Fig. 7: Tracking error comparison (scenario 2).

The uncertainty bounds adopted for the sampled parameters are summarized in Table V, with all parameters drawn from uniform distributions. The reported percentage variations are defined with respect to the nominal parameters of the LF model. The selected uncertainty ranges are consistent with the pre-mission knowledge that was available for small bodies such as Vesta, Bennu, and Eros prior to their respective exploration missions (e.g., [18]–[20]). The corresponding statistical results are reported in Table VI.

TABLE VI: Monte Carlo results

Metric	**Mean $\pm$ Std**
Mean Position Error	65.73 ± 27.12 m
Mean Velocity Error	$(6.48 \pm 2.84) \times 10^{-2}$ m/s

In scenario 2, the neural-network-based controller still achieves improved tracking accuracy with respect to the linear MPC (Fig. 7), while maintaining a comparable fuel consumption. However, the relative performance gain is less pronounced than in Scenario 1. This behavior can be attributed to the shorter prediction horizon and smaller sampling time adopted in this scenario. Under these conditions, the neural network predictions are closer to those obtained with the linear model, and the reduced time step allows the controller to react more promptly to deviations, mitigating tracking errors even in the presence of less accurate predictions.

D. Online learning

1) Data-driven learning: The simulations presented in this section consider a circular reference orbit whose initial state coincides with that adopted in scenario 1. The neural network is initially trained offline using the LF asteroid dynamical model. Fig. 8 illustrates the evolution of the reference orbit tracking accuracy during the online learning phase. The online refinement process is activated at $t = 1250$ min, allowing sufficient time for the buffer memory to accumulate representative trajectory data. To prevent degradation of control performance, an early-stopping mechanism is introduced. During online learning, the MPE is continuously evaluated based on the discrepancy between the reference trajectory and the actual spacecraft trajectory. If the MPE does not decrease

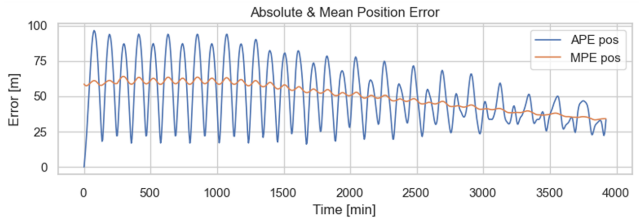


Fig. 8: Trajectory tracking error with online learning.

for a prescribed number of consecutive time steps, the online learning process is halted. This strategy mitigates overfitting to recent data and ensures stable closed-loop behavior. As a result, the MPE is reduced from 60.26 m when using the offline-trained network to 34.22 m after online learning, together with a noticeable reduction in error oscillations. The benefits of online learning become increasingly pronounced as the mismatch between the on-ground model and the true environment grows. In particular, when considering a 40% mismatch in the asteroid density ρ, a 6% mismatch in the angular velocity Ω, and a spin-axis tilt error of 6°, the MPE is reduced from 156 m to 72 m.

2) Physics-informed learning: To evaluate the impact of augmenting the online learning loss function with a physics-informed term, a test case characterized by a significant mismatch between the asteroid dynamical parameters estimated on ground and their true values is considered. Specifically, a 50% reduction in the asteroid density ρ, a 10% reduction in the angular velocity Ω, and a 12° error in the spin-axis tilt are introduced. During the initial phase of the simulation, up to $t = 1250$ min, online estimation of the dynamical parameters is performed using the approach described in Section II-F. In this preliminary assessment, the coefficients of the spherical harmonic expansion used to approximate the gravitational potential are assumed to be known. Once parameter refinement is completed, online learning of the neural network is activated. As shown in Fig. 9, the inclusion

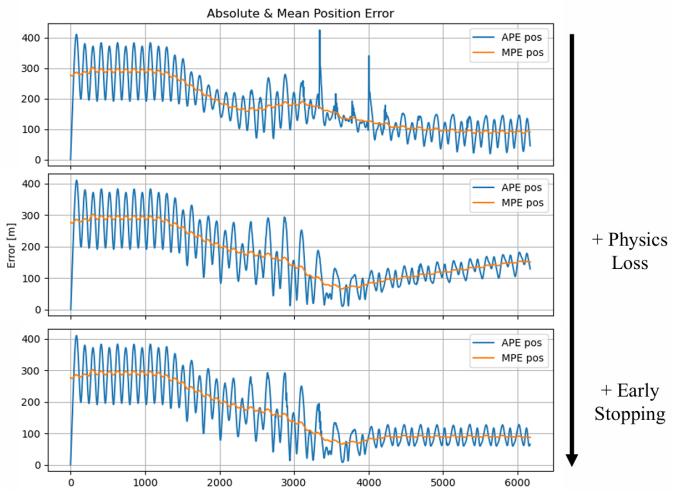


Fig. 9: Trajectory tracking error comparison during online learning with and without physics-informed loss.

of the physics-informed loss term has a stabilizing effect on the online refinement process, enabling the controller to reach the same final MPE as the purely data-driven approach, while achieving a faster convergence rate. These results are obtained through a careful tuning of the relative weights associated with the data-driven and physics-informed loss components. Further work is required to improve the robustness of this approach and to systematically assess its benefits.

E. Processor-in-the-Loop (PIL)

For the results reported in Table VII, the reference trajectory considered is a circular reference orbit whose initial state coincides with that adopted in scenario 1. In this set of experiments, the online learning procedure employs only the data-driven loss component. To clearly characterize the computational performance of the proposed control algorithm, the execution time is analyzed by decomposing the algorithm into three main components: neural network inference, solution of the optimization problem, and online training of the neural network.

TABLE VII: Algorithm execution times (PIL)

Execution Time [ms]	**Laptop**	**NVIDIA Jetson**
NN Inference	Mean: 3.255 Max: 40.308	Mean: 12.704 Max: 62.226
Optimization	Mean: 1063.220 Max: 3416.868	Mean: 3576.211 Max: 8508.476
Online NN Training	Mean: 181.178 Max: 265.413	Mean: 469.040 Max: 750.569

The duty cycle was evaluated from the mean execution time of two configurations of the control algorithm: optimization only and optimization combined with online neural network learning. The considered scenario features a control task period of 5 minutes, over which the duty cycle is defined. An execution time threshold of 10% of the task period is assumed as an acceptable upper bound for on-board feasibility.

TABLE VIII: Duty cycle of the control algorithm executed on the NVIDIA Jetson

Task	**Duty Cycle [%]**
Optimization	1.19
Optimization + Online Learning	1.35

IV. Conclusions

This work presented a hybrid control framework that combines a Transformer-based neural network architecture with Model Predictive Control to enable autonomous spacecraft proximity operations around small bodies. The neural network has been shown to effectively capture the complex gravitational dynamics of the asteroid and to reliably predict future spacecraft trajectories given the initial state and applied control inputs. It is worth noting that this capability is demonstrated in the vicinity of a reference orbit. Future studies should extend the analysis to more complex reference trajectories and a wider range of operational regimes. By leveraging the neural network

as a surrogate dynamics model within the MPC framework, the proposed approach achieves improved orbit tracking accuracy and fuel efficiency compared to a linear MPC baseline, as expected given the enhanced modeling fidelity. At the same time, it provides comparable tracking performance to a classical nonlinear MPC, while significantly reducing the computational burden, since the latter requires repeated integration of a complex nonlinear motion equations within the optimizer. The advantages of the proposed framework are mainly evident for sufficiently large prediction horizons and sampling times; when these are significantly reduced, the discrepancy between linear and nonlinear models becomes less pronounced, and the increased controller reactivity mitigates modeling errors. The online refinement of the neural network further enhances control performance, particularly in the presence of significant mismatches between the on-ground dynamical model and the true small-body environment. The introduction of a physics-informed term within the online learning process shows promising potential to guide training and accelerate convergence, especially under large parameter uncertainties. However, the current implementation is not yet fully robust, as the contribution of the physics-informed component must be carefully weighted to avoid degrading learning performance. Further investigation is therefore required to improve the stability and reliability of this approach. Finally, Processor-in-the-Loop experiments indicate that the proposed framework does not incur a significant degradation in execution time, supporting its suitability for onboard implementation on resource-constrained platforms.

REFERENCES

[1] E. F. Camacho and C. Bordons, *Model Predictive Control*, 2nd ed. London, U.K.: Springer, 2007.

[2] L. Cheng, Z. Wang, Y. Song, and F. Jiang, "Real-time optimal control for irregular asteroid landings using deep neural networks," *Acta Astronautica*, vol. 170, pp. 66–79, 2020.

[3] J. Park, M. R. Babaei, S. A. Munoz, A. N. Venkat, and J. D. Hedengren, "Simultaneous multistep transformer architecture for model predictive control," *Computers & Chemical Engineering*, 2023.

[4] Y. Gu and D. Li, "A transformer-based surrogate model predictive control for complex chemical processes," in *Proc. China Automation Congress (CAC)*, 2024.

[5] D. Celestini, D. Gammelli, T. Guffanti, S. D'Amico, E. Capello, and M. Pavone, "Transformer-based model predictive control: Trajectory optimization via sequence modeling," *IEEE Robotics and Automation Letters*, 2024.

[6] L. Capra, A. Brandonisio, and M. R. Lavagna, "Reinforced model predictive guidance and control for spacecraft proximity operations," *Aerospace*, vol. 12, no. 9, p. 837, Sep. 2025, doi: 10.3390/aerospace12090837.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 2017, pp. 5998–6008.

[8] A. Colagrossi, "Coupled dynamics around irregularly-shaped bodies with enhanced gravity field modelling," M.S. thesis, Dept. Aerosp. Sci. Technol., Politecnico di Milano, Milan, Italy, 2015.

[9] D. K. Yeomans, P. G. Antreasian, J.-P. Barriot, S. R. Chesley, D. W. Dunham, R. W. Farquhar, J. D. Giorgini, C. E. Helfrich, A. S. Konopliv, J. V. McAdams, J. K. Miller, W. M. Owen, D. J. Scheeres, P. C. Thomas, J. Veverka, and B. G. Williams, "Radio science results during the NEAR-Shoemaker spacecraft rendezvous with Eros," *Science*, vol. 278, no. 5346, pp. 2106–2109, Dec. 2000.

[10] H. D. Curtis, *Orbital Mechanics for Engineering Students*, 3rd ed. Oxford, U.K.: Butterworth-Heinemann, 2014.

[11] NASA Planetary Data System, "NEAR collected shape and gravity models," Planetary Data System, 2026. [Online]. Available: https://sbn.psi.edu/pds/resource/nearbrowse.html

[12] Z. Zhao, X. Ding, and B. A. Prakash, "PINNsFormer: A Transformer-Based Framework for Physics-Informed Neural Networks," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2024.

[13] J. Martin and H. Schaub, "Physics-informed neural networks for gravity field modeling of the Earth and Moon," *Celestial Mechanics and Dynamical Astronomy*, vol. 134, no. 5, pp. 1–24, 2022.

[14] COIN-OR Foundation, "IPOPT: Interior Point OPTimizer," [Online]. Available: https://coin-or.github.io/Ipopt/

[15] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.

[16] W. M. Kaula, *Theory of Satellite Geodesy: Applications of Satellites to Geodesy*. Waltham, MA, USA: Blaisdell Publishing Company, 1966.

[17] A. Pellacani, M. Graziano, and M. Suatoni, "Design, development, validation and verification of GNC technologies," in *Proc. European Conf. for Aeronautics and Space Sciences (EUCASS)*, Madrid, Spain, 2019, doi: 10.13009/EUCASS2019-38.

[18] S. R. Chesley, D. Farnocchia, M. C. Nolan, D. Vokrouhlický, P. W. Chodas, A. Milani, F. Spoto, B. Rozitis, L. A. M. Benner, *et al.*, "Orbit and bulk density of the OSIRIS-REx target asteroid (101955) Bennu," *Icarus*, vol. 235, pp. 5–22, May 2014, doi: 10.1016/j.icarus.2014.02.020.

[19] D. K. Yeomans, P. G. Antreasian, A. Cheng, D. W. Dunham, R. W. Farquhar, R. W. Gaskell, J. D. Giorgini, C. E. Helfrich, A. S. Konopliv, J. V. McAdams, J. K. Miller, W. M. J. Owen, P. C. Thomas, J. Veverka, and B. G. Williams, "Estimating the mass of asteroid 433 Eros during the NEAR spacecraft flyby," *Science*, vol. 285, no. 5427, pp. 560–564, Jul. 1999, doi: 10.1126/science.285.5427.560.

[20] M. D. Rayman and R. A. Mase, "Dawn's exploration of Vesta," *Acta Astronautica*, vol. 94, no. 1, pp. 303–314, Jan. 2014, doi: 10.1016/j.actaastro.2013.08.003.