M. Pugliatti, A. Scorsoglio, R. Furfaro, F. Topputo
Onboard State Estimation Around Didymos with Recurrent Neural Networks and Segmentation Maps
IEEE Transactions on Aerospace and Electronic Systems, published online 22/06/2023
doi:10.1109/TAES.2023.3288506

Onboard state estimation around Didymos with recurrent neural networks and segmentation maps

MATTIA PUGLIATTI
Politecnico di Milano, Milan, Italy

ANDREA SCORSOGLIO
University of Arizona, Tucson, AZ, USA

ROBERTO FURFARO
University of Arizona, Tucson, AZ, USA

FRANCESCO TOPPUTO
Politecnico di Milano, Milan, Italy

***Abstract*— When considering the proximity environment of a small body, the capability to navigate around it is of paramount importance to enable any onboard autonomous decision-making process. Onboard optical-based navigation is often performed by coupling image processing algorithms with filtering techniques to generate position and velocity estimates, providing compelling navigation performance with cost-effective hardware. These same processes could be addressed with data-driven ones, at the expense of a sufficiently large dataset. To investigate to what extent can these methods substitute traditional ones, in this paper we develop a possible onboard methodology based on segmentation masks, convolutional extreme learning machine architectures, and recurrent neural networks to respectively generate simpler image inputs, map single-frame data into position estimates, and process multiple-frame position sequences to generate both position and velocity estimates. Considering the primary of the Didymos binary system as a case study and the possibility to complement optical observations with LiDAR data, we show that recurrent neural networks would bring only limited improvement in position reconstruction for the case considered while they would be beneficial in estimating the velocity, especially when considering complementing LiDAR data.**

***Index Terms*— Image Processing, Navigation, Recurrent Neural Networks, Segmentation, Small-Bodies**

Manuscript received XXXXX 00, 0000; revised XXXXX 00, 0000; accepted XXXXX 00, 0000.

M.P and F.T would like to acknowledge the funding received from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 813644. *(Corresponding author: Mattia Pugliatti)*

Mattia Pugliatti and Francesco Topputo are with the Department of Aerospace Science and Technology, Politecnico di Milano, 20156, Milan, Italy. (e-mail:mattia.pugliatti@polimi.it,francesco.topputo@polimi.it). Andrea Scorsoglio and Roberto Furfaro are with the Department of Systems and Industrial Engineering, University of Arizona, 85721, Tucson, Arizona. (e-mail:andreascorsoglio@email.arizona.edu, robertof@arizona.edu).

Acronyms

CELM	Convolutional Extreme Learning Machine
CNN	Convolutional Neural Network
CoM	Center of Mass
CPO	Close Proximity Orbit
GD	Gradient Descent
IC	Initial Condition
IP	Image Processing
KF	Kalman Filter
LiDAR	Light Detection And Ranging
LoS	Line of Sight
LSTM	Long-Short Term Memory
RNN	Recurrent Neural Network
UNet	U-shaped Network

I. INTRODUCTION

Missions towards small bodies, such as asteroids and comets, are becoming increasingly interesting for national space agencies, companies, and smaller players such as research centers and universities [1]. The capability to autonomously navigate around a known celestial body is of paramount importance to enable any autonomous decision-making process onboard a spacecraft [1]. When considering the proximity environment of a small body and all the navigation sensors available on the market, cameras are usually preferred as they are light, compact, and have low power demand. For these reasons, the use of passive cameras, in combination with Image Processing (IP) algorithms, provides compelling navigation performance with cost-effective hardware.

Several IP algorithms have been deployed in various exploration missions towards small bodies. Considering a close proximity scenario about a small body, two navigation regimes are discerned: mid and close-range. In the mid-range regime, the main optical feature used about the body is its global outline, while in the close-range regime, local features such as craters, boulders, and other surface characteristics are extracted and used.

In the mid-range regime, centroid algorithms are often considered a robust, easy-to-use baseline both for rendezvous [2], flyby [3], and deflection [4] missions. The simplicity of these algorithms generally comes at the cost of low performance, especially when considering high-phase angles and highly irregular shapes [5]. To overcome these limitations, over the years modified centroid algorithms have been developed that adopt scattering functions [6], [7]. Another approach that can be used to cope with high phase angles is represented by centroid and apparent diameter algorithms. A very well-known algorithm is the one introduced in [8], which develops a non-iterative ellipse fitting to retrieve the position with respect to a body modeled as a tri-axially ellipsoid. Other important works are the one that uses limb fitting [5] or Lambertian sphere correlation algorithms [9]. However, these methods, originally designed for regular celestial objects such as planets and moons, are not well suited for applications to irregular small bodies.

In the close-range regime, features-based methods are usually preferred due to the abundance of resolved local features and higher performance. For example, Rosetta [10] adopted small-scale, high-resolution, digital models called maplets centered on specific landmarks of interest, scattered across the surface of the target body as reference points. Combining together elevation, albedo, and photometric models each landmark's appearance is simulated at varying illumination conditions. By correlating on-ground these digital representations with their real counterpart extracted from images, the position of the spacecraft is reconstructed. A similar methodology is also developed for onboard application in the Osiris-Rex mission [11], exploiting natural features rendered and correlated thanks to detailed maps generated in previous phases of the mission. Another type of strategy is instead adopted by the Hayabusa 2 mission [12], which uses artificial landmarks left on the surface of the body for image processing and navigation during critical operations. On the other hand, Hera will deploy relative feature tracking algorithms in combination with navigation filters for the most critical phases in the vicinity of Didymos [9], [13]. Finally, there also exist IP algorithms that use dynamic triangulation [14], [15] exploiting features such as planets or local features on the surface of a body, and extracting triads of lunar craters and image invariant features with onboard catalogs [16], to generate position and velocity navigation solutions.

The majority of the methods introduced before can often provide only pinpoint position estimates, which alone do not contain velocity estimates. While estimating the velocity from the difference of subsequent position estimates could seem a viable solution, this is often very detrimental due to noise on the position estimates. To overcome this limitation and be able to reconstruct a full state vector of the spacecraft comprising both position and velocity components, these IP algorithms are often coupled with navigation filters (often variations of a Kalman Filter (KF)), which use a-priori information on the initial conditions and a dynamic model to generate sequential estimates onboard.

Within this context, both the onboard IP and KF algorithms can be substituted by promising data-driven methods performing both tasks at the same time. The IP can be performed with deep-learning architecture such as Convolutional Neural Network (CNN) and its variants, which have demonstrated their exceptional capability to extract high-level features from images and process their non-linear mapping with labels, representing the state of the art in computer vision for several tasks [17]. On the other side, a KF can be substituted by recurrent architectures that are specifically designed to process time series. In a Recurrent Neural Network (RNN) the full dynamic or the state transition matrix is implicitly substituted by the Long-Short Term Memory (LSTM) cells within the recurrent architecture. The transition from one state to the next is learned through data, which is a drawback of this approach. The architecture needs to learn the proper representation and a simple change in the dynamic model requires the generation of a new dataset and a fine-tuning of the network.

Starting from these considerations, this work is supported by three streams of existing literature and research previously analyzed by the authors and others: the use of small bodies segmentation maps for navigation, the role of RNN to improve single point estimates, and the design and training of efficient networks for IP.

The first one is represented by the generation and use of segmentation maps for optical-based navigation using visible images. In the general computer vision domain, image segmentation is an active field of research. In this work, following the definition in [17], segmentation is defined as the task that performs per-pixel classification on an image. A good overview of the most important techniques can also be found in [17]. In the current state of the art, deep architectures such as those described in [18] or in [19] seems to be the preferred choice [17] for the most complex tasks. In [18] a successful architecture referred to as U-shaped Network (UNet) is introduced to perform segmentation for biomedical applications. The success of this architecture pivots on two aspects: a symmetrical architecture that lacks the fully connected layer between encoding and decoding stacks and the idea to concatenate copies of trained encoder layers on the decoder to retain feature representation. Moreover, in [18] the authors successfully demonstrate the capability to train the network with a very limited dataset constituted only of 30 images, largely relying on data-augmentation techniques to virtually increase the size of the training set. Another popular architecture is the one introduced in [19], which represents a natural evolution of the UNet, making use of less memory by avoiding the storage of the entire pooling layers and instead opting for the storage of the indices.

Considering image segmentation for space applications, a variety of methods and applications have been investigated over the past decades to improve the autonomous perception capabilities of interplanetary missions. In [20], image segmentation is used to classify geological properties of the terrain while in [21] to identify plumes and jets from comets and moons. In [22], a series of advanced image processing techniques, amongst which image segmentation, are demonstrated for the enhancement of the scientific return of flyby missions around small bodies. More recently, the simple implementation and easy access to pre-trained UNet and encoder architectures have encouraged a proliferation of deep networks for image segmentation in space applications. For example, UNets are used in [23], [24] to obtain hazard maps of the Moon or of small body surfaces [25] to perform safe landing site selection. The works of [26] and [27] offer comprehensive overviews of the performance of segmentation architectures applied to the surface of small bodies and for landing site selection of planetary surfaces respectively. Moreover, several works [24], [25], [28] are now introducing uncertainty quantification metrics [29], [30] that accompany the generation of segmentation maps to increase robustness and applicability for future missions.

Finally, of major relevance for this paper is the work illustrated in [28], where a methodology is presented that can be used to perform semantic segmentation of small-body surfaces based on morphological features such as craters, boulders, and terminator regions. This is done with the use of a UNet CNN architecture, which can be coupled onboard with other CNN for optical navigation, as it is illustrated in [31]. In the latter, using segmentation masks of two different asteroids, a CNN is tasked to classify an approximate position in space around the body while a Normalized Cross-Correlation algorithm is used with an iterative procedure to further refine the approximate position provided by the CNN. In [31] is also observed that the prediction with regular shapes is relying heavily on the geometric disposition of features such as boulders and craters. On the other hand, the prediction with irregular shapes is more reliant on the characteristics of the global outline. From this perspective, when using segmentation maps, regular bodies represent a more challenging case study than irregular ones.

The second one is characterized by the role of RNN architectures to improve the performance of single-frame classification methods. This has been exemplarily shown in [32], in which a CNN is tasked with the classification of the location of an image from a variable resolution grid of classes around the Earth. The approach presented by the authors demonstrated to be capable to perform adequate localization with a CNN, as well as being able to improve performance by 50% when considering sequences of images analyzed with a RNN. Such work has been inspirational in [31], which attempted to replicate the same idea but on a regular grid of points around different asteroids and by using segmentation maps. From these works as well as from the traditional onboard usage of KF arises the interest to investigate to what extent sequences of images can be used instead of single frames to improve position reconstruction performance and complement them with velocity estimates.

The third and last stream is represented by Convolutional Extreme Learning Machine (CELM) networks against CNN ones. The latter often relies on Gradient Descent (GD) methods for training, which requires substantial computational resources, that in turn inhibit an effective exploration of the architecture design space. In this context, Extreme Learning Machine [33], [34], [35] offers an alternative strategy and a new paradigm that speedup training time while relying on random neurons and simpler architectures. This concept is extended further for computer vision tasks in [36], in which the concept of CELM is introduced. The interested reader is directed to [37] for a recent systematic literature review about CELM. The core idea of CELM theory is that hierarchical pooling architectures with randomized kernels are inherently capable to extract spatial information from images [38] better or at the same performance level as CNN whose kernels have been trained via GD methods. Training in a CELM happens by solving a regularized Least Square (LS) problem, which adjusts the weights β of the connection between the two last layers in the network to match training and validation data with network predictions. Because training happens with a LS method, it is orders of magnitude faster than GD. Moreover, since segmentation maps represent a more comprehensive, yet simpler representation of the environment with respect to grayscale images, simpler architectures could be inherently more suitable as IP methods. For similar reasons, the work presented in [39] compares the performance of position estimation with CNNs and CELMs around four different shape models and with different labeling strategies. It is found that for Didymos the performance of a CNN and the one of a CELM are comparable. The findings in such work have been instrumental for the assumptions under which CELMs architectures are designed in this paper.

Pivoting on the work performed in these three research areas, in this paper the authors investigate to what extent can RNNs be used to refine position estimates taking as input multiple frames rather than single ones and whether RNNs can be used to determine the velocity components of the spacecraft state vector. The authors prove that the improvement on the position estimate is negligible, while velocity is estimated accurately only when considering rangefinder Light Detection and Ranging (LiDAR) data to decrease the error in the radial direction. To do that, a variety of trajectories around the primary body of the Didymos binary system is considered as a case study, since its interest for future exploration missions [40].

Finally, it is commented that with respect to traditional IP methods and navigation filters, the data-driven architectures presented in this work present both advantages and disadvantages from an operational point of view. Generalized data about the target body needs to be prepared before arrival and needs to be constantly fine-tuned at different stages of the mission to improve the performance and stability of the networks. These need to be trained on-ground using such data, but can then be uplinked easily as an architecture and a collection of optimal weights and biases. This would not be very different from the type of human-in-the-loop effort such as in [10], [41], [11], focusing entirely on the dataset generation and network training other than the algorithms re-design and fine-tuning. One drawback of the methodology is explainability in terms of explainability of the predictions, which is an active area of research.

The rest of the paper is organized as follows. In Section II the generation of the datasets, as well as the design and training procedure used for the CELM and RNNs, are illustrated in detail. The performance of these networks are then compared in Section III both for what concerns the capability to improve the position estimate as well as the one to generate a velocity estimate. Finally, in Section IV conclusions and future improvements are briefly discussed.

II. METHODOLOGY

To obtain the position and/or velocity estimates from a RNN, a training methodology is designed and divided into two main parts, as illustrated in Fig. 1. In the first one, CELM architectures are trained to generate position estimates using segmentation maps of Didymos as input. The best architecture is selected and used in inference on a vast test dataset to generate sequences of position estimates, which constitutes the input for the training of the RNN. In the second part, an ensemble of RNN is designed and trained to take as input variable sequences of the position estimates previously generated with the best CELM architecture with the goal of either improving the same position estimate or generating a velocity estimate. The sequence of position estimates generated with the CELM can either use only optical observables or complement them with rangefinder LiDAR data. After a sequence of N position vectors is obtained onboard, the RNN is used to analyze it and produce an estimated state vector valid for the N_{th} instance. This is either made up of only position (p, pl), velocity (v, vl), or both (pv, pvl) components. The l indicates whether or not LiDAR data has been used to generate the position with the CELM.

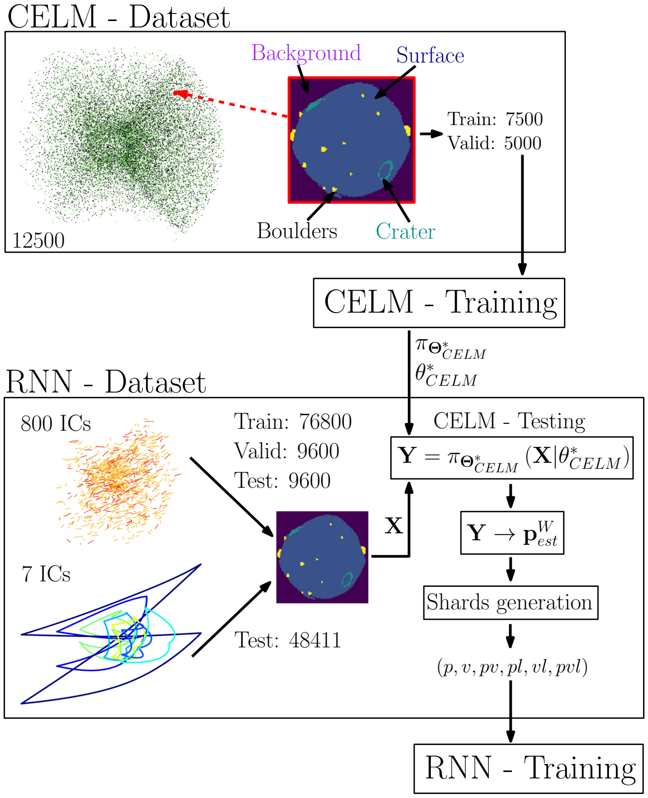


Fig. 1. Sketch of the combined training of the CELM and RNN methods.

In inference the CELM and RNN are applied sequentially after a first passage by a UNet that is used to generate the segmentation masks from visible grayscale images [28], [31]

In the schematic in Fig. 2, the networks of interest in this paper are represented as a function π which generates an output matrix $\mathbf{Y}$ as a function of the input $\mathbf{X}$, parameterized by the weights and biases θ and the hyper-parameters $\mathbf{\Theta}$. The optimal sets of θ and $\mathbf{\Theta}$ are denoted by a * superscript and are found via the training and validation methodology described in detail in the next sections.

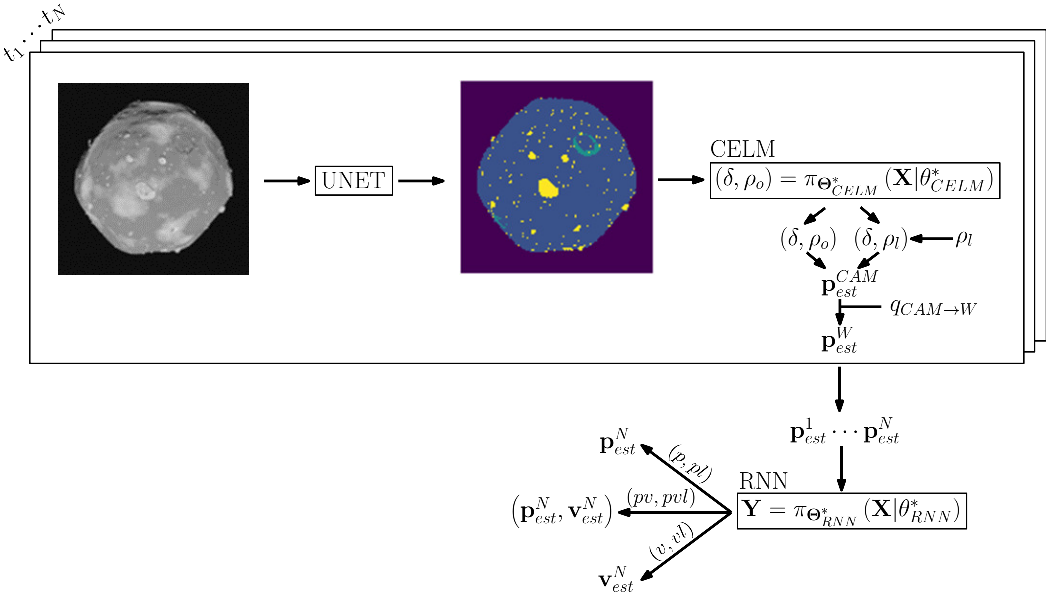
A. Dataset of the Convolutional Extreme Learning Machine architectures

Following the same procedure illustrated in [28], a dataset of 12500 segmentation maps around the primary of the Didymos binary system is generated in Blender[1]. Each segmentation map is composed of 1024×1024 pixels, each representing a specific morphological feature with a value from 0 to 3: Background (0), surface (1), craters (2), and boulders (3). The maps are obtained using the *Cycles* rendering engine and assigning a pass index to each layer. The dataset is split into training and validation sets respectively made of 7500 and 5000 samples.

All the 12500 maps are uniformly distributed in spherical coordinates across a region around Didymos with range $5 < \rho < 30$ km, azimuth $-85 < \phi_1 < 85$ deg, and elevation $-50 < \phi_2 < 50$ deg in what is defined as the W reference frame. The range interval is chosen to always have the body resolved by the field of view of the simulated sensor, which is 10×10 deg wide. The W reference frame is an inertially fixed reference frame, centered on the Center of Mass (CoM) of the Didymos asteroid, with the X-axis oriented towards the Sun and the Z-axis as the north pole of the body [40]. This frame is used since it simplifies the couplings between geometric viewpoints around the body and illumination conditions, as the Sun is always positioned towards the X-axis. Also, the characteristic of these points is chosen as a reasonable assumption for a realistic close-proximity scenario and is based on previous experience gained by the authors on the design of the close-proximity operations of the Milani mission [40] and analogies with the Hera mission [13]. Taking these considerations, the authors assume a scenario about the Didymos asteroid is realistic if it develops above the primary asteroid at various ranges and with low-medium phase angles to allow scientific acquisitions with passive sensors. It is also noted that the absolute value of elevation from the equatorial plane of the asteroid is kept below 50 deg, thus excluding trajectories passing directly above the asteroid poles. This choice has been inspired by the trajectories of Hera [13] and Milani [40] which do not traverse the polar regions of Didymos for most of their operational lives.

The segmentation maps are rendered assuming ideal pointing and are accompanied by a set of labels that

[1] https://www.blender.org/, last time accessed: 3rd of May 2022.



$t_1 \cdots t_N$
UNET
CELM
$(\delta, \rho_o) = \pi_{\Theta^*_{CELM}}(\mathbf{X}|\theta^*_{CELM})$
(δ, ρ_o)
(δ, ρ_l)
ρ_l
$\mathbf{p}^{CAM}_{est}$
$q_{CAM \to W}$
$\mathbf{p}^{W}_{est}$
$\mathbf{p}^{1}_{est} \cdots \mathbf{p}^{N}_{est}$
RNN
$\mathbf{Y} = \pi_{\Theta^*_{RNN}}(\mathbf{X}|\theta^*_{RNN})$
$\mathbf{p}^{N}_{est}$
(p, pl)
$(\mathbf{p}^{N}_{est}, \mathbf{v}^{N}_{est})$
(pv, pvl)
(v, vl)
$\mathbf{v}^{N}_{est}$

Fig. 2. Sketch of the combined CELM and RNN methods in inference.

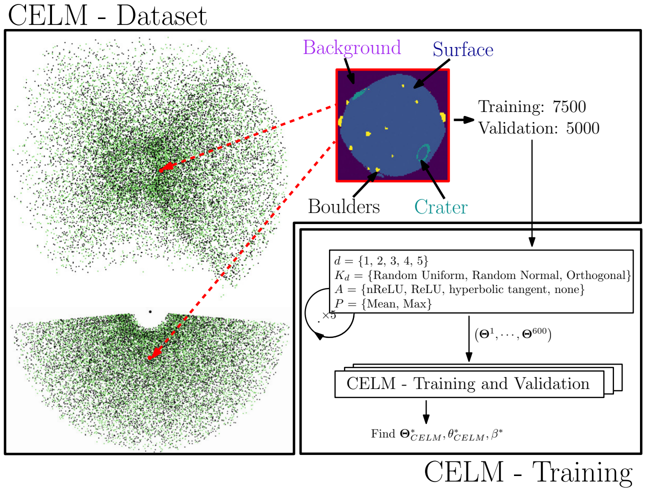
can be extracted from the image and can be used to establish the position of a spacecraft w.r.t the asteroid. Pivoting on the findings illustrated in [39], in this work the strategy based on optical observables is adopted. The CELM is therefore tasked to generate a 3-values vector that represents quantities related to properties identified in the image:

- The first two components represent the estimated correction in pixel in the image plane between the center of brightness and the projected CoM. This quantity is also referred to as δ or scattering correction.
- The third component is the range ρ from the CoM of Didymos.

Note that δ is translation invariant and, if properly trained, can be predicted with random camera boresight rotation to be rotation invariant. Using δ as a label a network can be trained to 'center' an small body using its images as input. Using the same image preprocessing pipeline described in detail in [39] (without the inclusion of noise in the segmentation maps), data augmentation is performed on the input, and each map-label pair is transformed into a 128 × 128 matrix with the asteroid not necessarily appearing centered in the frame but rather randomly displaced in it.

B. Training of the Convolutional Extreme Learning Machine architectures

Input and labels are normalized and then used to train 600 different convolutional architectures using the CELM training paradigm [36], [39]. This consists essentially in correcting only the connection between the last two layers of the architecture by finding an optimal set of weights β using a regularized LS method and random weights and biases in the kernels used within the convolutional layers. The training methodology of the CELM is illustrated in the schematic in Fig. 3 and is the same used in [39]. The training and validation datasets are also represented in Fig. 3 respectively by black and green points in position phase space around the Didymos asteroid. From these points, segmentation maps are generated and used to train and validate the CELM architectures with varying hyper-parameters.



CELM - Dataset
Background
Surface
Training: 7500
Validation: 5000
Boulders
Crater
$d = \{1, 2, 3, 4, 5\}$
$K_d = \{\text{Random Uniform, Random Normal, Orthogonal}\}$
$A = \{\text{nReLU, ReLU, hyperbolic tangent, none}\}$
$P = \{\text{Mean, Max}\}$
×5
$(\mathbf{\Theta}^1, \cdots, \mathbf{\Theta}^{600})$
CELM - Training and Validation
Find $\mathbf{\Theta}^*_{CELM}, \theta^*_{CELM}, \beta^*$
CELM - Training

Fig. 3. Sketch of the first part of the training with the CELM.

The details of the hyper-parameters tested in the 600 different CELM architectures are illustrated in the Appendix for the interested reader. Such vast exploration of the architecture design space of the CELM is possible only because training is orders of magnitude faster than the one of CNN [38], [36], [37], [39], since the optimization scheme to orient the weights and biases of the network is not based on GD methods but LS ones and do not require backward passes and iterations.

The best CELM design is thus found as the one achieving the minimum positioning error on the validation set and referred to as $\pi_{\Theta^*_{CELM}}$. This training framework can be adapted for a real mission case to make efficient use of the resources available for training and reduce costs. The combination of exploration with a first pass to find the best set of possible architectures and exploitation with a second one dedicated to the optimization of their weights and biases ultimately guarantees an efficient and robust development of a network.

Using the output of the CELM, the position is generated following the same procedure illustrated in [39], here briefly described. The $\boldsymbol{\delta}$ is transformed into a LoS (Line of Sight) vector in the CAM reference frame. Using the attitude quaternion of the spacecraft (which is assumed to be known from attitude determination from a Star-tracker) and assuming to know the rigid rotation between the inertial reference frame used by the Star-tracker and a known asteroid frame, this Line of Sight is transformed with the use of range ρ into a position estimate in W reference frame. Note that in this paper no error is simulated on the attitude quaternion so an ideal measurement model from the Star Tracker is used. The estimated position in W frame is thus computed as:

$$\boldsymbol{p}^{W}_{est} = \boldsymbol{q}_{CAM \to W} \cdot \boldsymbol{p}^{CAM}_{est} \tag{1}$$

where $\boldsymbol{p}^{CAM}_{est}$ is the estimated position in the camera frame and $\boldsymbol{q}_{CAM \to W}$ is the quaternion that rotates from the CAM to W frame. In this paper, ρ is either estimated from the images with the best CELM architecture (ρ_o) or with the use of a rangefinder LiDAR sensor (ρ_l). The latter is simulated with the addition of normally distributed noise on the true range ρ_t between CoM of Didymos and the spacecraft as:

$$\rho_l = \rho_t + \sigma_l \cdot \Omega \tag{2}$$

where Ω is a normal random distribution function, and σ_l is the LiDAR standard deviation measured as $\sigma_l = \sqrt{\sigma_h^2 + \sigma_s^2}$, σ_h being the contribution by the instrument uncertainty (assumed to be 1 m as an educated guess from the DLEM rangefinder LiDARs from Jenoptik [2]) and σ_s the uncertainty provided by the deviation of the small-body shape ($\sigma_s = 13.63$ m) due to irregularities from a sphere centered in the CoM with a radius equal to the mean value of 392.48 m.

C. Dataset of the Recurrent Neural Networks

Using the dynamical model illustrated in [40] and the propagator tool developed to design the Close Proximity Orbit (CPO) of Milani, a dataset of position-velocity pairs is generated. The dynamical model considered three main accelerations: the gravity of Didymos, the third-body effect of the Sun, and the solar radiation pressure. Differently than the model in [40], the gravitational acceleration caused by Dimorphos, the secondary body of the Didymos binary system, is not modeled since the focus of this work is set on the primary. The gravity of Didymos is modeled as a point mass for regions above 1.1 km and using a polyhedra model below this range. This threshold is clearly illustrated in the perturbation analysis in [40]. Also, instead of having the position of Didymos and the Sun resolved precisely using ESA's Hera mission kernels as in [40] at any given time, both Didymos and the Sun positions are assumed at a fixed epoch to simplify the analysis, and allow an adequate comparison between the different CPOs.

In such an environment, trajectories are designed using a strategy consisting of ballistic arcs patched together at maneuver points, called waypoints. Each ballistic arc between two consecutive waypoints is based on a stepwise differential correction procedure and is the result of an iterative targeting problem. First, initial conditions are set that determine the ballistic trajectory to be flown by the spacecraft. These include the position of the two waypoints, the initial epoch, and the time of flight between waypoints. Second, a restricted two-body problem Lambert's solver is used to find a suitable first guess solution for the initial velocity. Third, the initial state is propagated forward using the dynamical model. The deviation between the actual endpoint and the desired one is used to compute a correction term on the initial velocity generated at the previous step. The last step is iterated until the discrepancy between the two end states is below an arbitrarily defined threshold. More details about the differential correction scheme can be seen in [40].

Using such scheme in combination with the dynamical model, a total of 144411 trajectory points scattered across Didymos are generated. These are divided into two main groups, made of 96000 and 48411 points. The first comprises short pieces of open trajectories computed from 800 Initial Condition (IC)s randomly distributed across the position and velocity phase space, as illustrated in Fig. 4 and Fig. 5. These trajectories are obtained from a forward propagation with a fixed timestep of 150 s for a total of 120 steps. This randomly distributed dataset is divided into training, validation, and test splits respectively made of 76800, 9600, and 9600 samples. Note that the splits are divided consistently in different random trajectories so that 640, 80, and 80 ICs are used for each split. The second dataset comprises closed trajectories from 7 different ICs, which include intermediate maneuvers. Closed trajectories are obtained by simply constraining the first and last waypoints to be the same, using the differential correction scheme used in [40] and described before. This dataset is made of 48411 points which are entirely used for testing. The CPOs described in this dataset are representative of possible geometries to be used in the proximity of a small body, and in particular, are representative of real CPOs which can be adopted

[2] https://www.jenoptik.com/products/lidar-sensors-technologies/laser-rangefinders/oem-modules-system-integration/dlem, last accessed 27th of October 2022.

around the Didymos asteroid [40]. The geometries are arbitrarily chosen from experience gained by the authors on the Milani mission as well as loosely inspired from geometries seen in previous missions around small bodies such as in [41].

Once the position-velocity pairs have been determined for all datasets needed for the design of the RNN, segmentation maps are generated for each of the 144411 trajectory points. Each map is then used in inference for the best CELM obtained in the previous section to generate 144411×2 position estimates (with the use of ρ_o or ρ_l).

After this passage, the entire dataset used for training of the RNN is transformed into a sequence of fixed-interval position-velocity pairs expressed in the W reference frame. It is noted that computing the position estimate apriori greatly simplifies and speed-up the training of the RNN while retaining its capability to be developed for an onboard application. Ultimately, this choice has proven to be successful in enabling an extensive exploration of the architecture design space of RNN architectures without the problem to have to deal with a large amount of imagery data.

Each of the 807 different trajectories considered for the RNN is divided into multiple shards defined by 3 parameters: the time interval between position estimates ΔT (which is a multiple of 150 s), the total number of position estimates N from the CELM to be used to generate an estimate with the RNN, and the initial sample of the shard from the original trajectory j. In this paper, 5 possible combinations of (ΔT,N) are investigated, as illustrated in Tab. I, while j is rolled forward until there are enough points in the trajectory to generate a shard of N samples. For simplicity, the test sets are divided into Te_1 and Te_2 , respectively representing the random shards within the training envelope generated by the 80 ICs, and the ones of the 7 CPOs illustrated in Fig. 4 and Fig. 5.

Tab. I
Number of shards in each train, validation and test sets for each possible combination investigated.

ΔT	N	Train	Validation	Te_1	Te_2
150	5	74240	9280	9280	48303
150	30	58240	7280	7280	47628
1800	5	46069	5760	5760	47104
3600	5	15337	1920	1920	45796
5400	3	30685	3840	3840	46432

D. Training of the Recurrent Neural Networks

The RNN is designed after a thorough hyper-parameters search using a combination of LSTM cells and a single layer of neurons. The RNN takes as input a sequence of N previously estimated positions obtained with the best CELM identified in the first part of the training and produces the current estimate of the position, velocity, or both, with and without the use of LiDAR. The entire training procedure of the RNN is illustrated in 6.

Using the train and validation shards with the properties illustrated in Tab. I in combination with the six different labeling strategies (p, v, pv, pl, vl, pvl), an extensive hyper-parameter search for the definition of the best RNN architectures is performed. The details of the hyper-parameter search are illustrated in the appendix.

Repeating this methodology for each of the 6 labeling strategies considered in this work, best-performing architectures are obtained. The characteristics of the best performing RNN architectures found with the strategy illustrated in the appendix are summarized in Tab. II.

Tab. II
Best hyper-parameters $\boldsymbol{\Theta}^*$ of the RNN for each labeling strategy.

Label	$\mathbf{p}$	$\mathbf{v}$	$\mathbf{p}, \mathbf{v}$	$\mathbf{p}$	$\mathbf{v}$	$\mathbf{p}, \mathbf{v}$
LiDAR	No	No	No	Yes	Yes	Yes
Name	R_p	R_v	R_{pv}	R_{pl}	R_{vl}	R_{pvl}
ΔT $[s]$	150	3600	3600	150	3600	3600
N	30	5	5	30	5	5
lr	10^{-2}	10^{-3}	$2 \cdot 10^{-4}$	10^{-3}	10^{-3}	10^{-3}
m	128	256	128	64	256	64
n_{lstm}	16	128	128	256	512	256
n_{ne}	16	64	512	32	512	256

III. RESULTS

In this section, the results of the position and velocity onboard reconstruction are analyzed. For a better interpretation, a consistent color scheme and notation are adopted. The performance of the RNNs are characterized consistently by a palette of blue colors of the *jet* colormap while the CELMs ones by red colors of the same colormap, as it is possible to see from the legends in Fig. 7 and Fig. 8. Also, the performance related to the CELM are annotated as C_i, the ones related to the RNN as R_i, the i representing the labeling strategy. To assess the onboard state reconstruction capabilities, the following global metrics are defined:

$$\varepsilon_p = \frac{\|\boldsymbol{p}_{est} - \boldsymbol{p}_{true}\|_2}{\|\boldsymbol{p}_{true}\|_2} \cdot 100 \tag{3}$$

$$\varepsilon_v = \frac{\|\boldsymbol{v}_{est} - \boldsymbol{v}_{true}\|_2}{\|\boldsymbol{v}_{true}\|_2} \cdot 100 \tag{4}$$

where $\boldsymbol{p}_{est}$ and $\boldsymbol{v}_{est}$ are the estimated position and velocity vectors, $\boldsymbol{p}_{true}$ and $\boldsymbol{v}_{true}$ are the true position and velocity vectors.

In Tab. III it is possible to see a summary of the performance of the various methods with different labeling strategies. In particular, it is possible to see that the coupling between LiDAR data and optical observables is beneficial in terms of performance, which are one order of magnitude better compared to estimates generated with optical observables alone. It is also noted that the RNNs reconstruct a better estimate whenever they are solely focused on the position or velocity, as in the p, v, pl, and

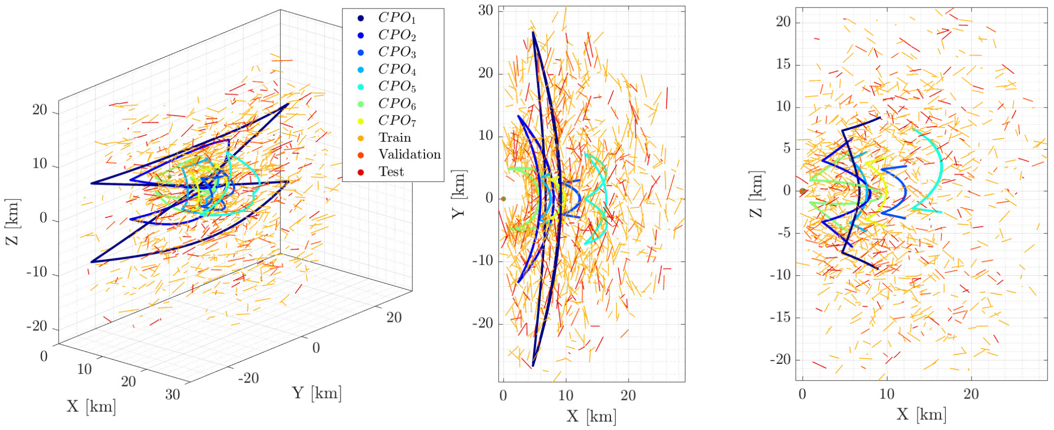


Fig. 4. Train, validation, and test sets of the RNN in the position phase space. All CPOs are used for testing.

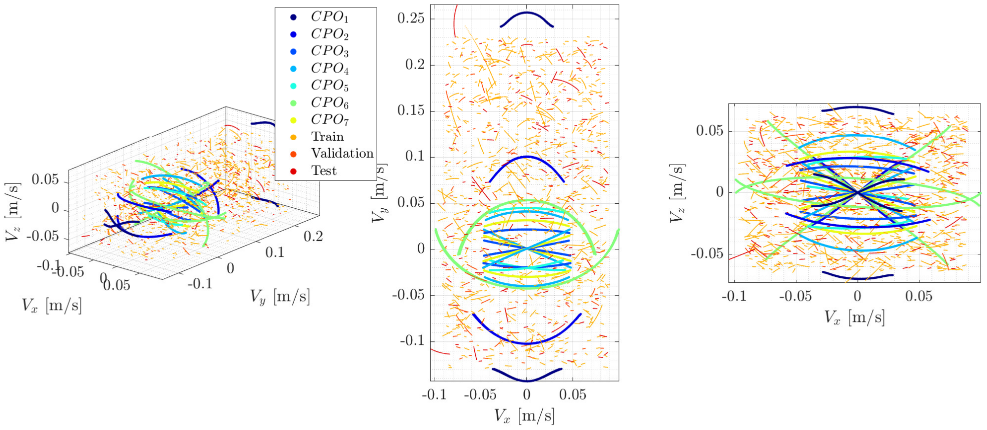


Fig. 5. Train, validation, and test sets of the RNN in the velocity phase space. All CPOs are used for testing.

vl cases, and not in the mixed labeling strategies such as in the pv and pvl cases.

Tab. III
Performance of the state reconstruction by CELM (C) and RNN (R) with different labeling strategies. The values expressed are the $\mu(\varepsilon_p)$ or $\mu(\varepsilon_v)$ and their corresponding (σ).

Labels	Dataset	C	R	R
p	Te_1	2.69 (2.15)	1.28 (1.25)	-
	Te_2	2.70 (2.01)	1.30 (0.87)	-
v	Te_1	-	-	31.86 (27.75)
	Te_2	-	-	33.67 (29.29)
pv	Te_1	2.67 (2.05)	2.74 (1.66)	34.46 (28.82)
	Te_2	2.69 (2.00)	2.74 (1.67)	41.16 (33.12)
pl	Te_1	0.24 (0.21)	0.22 (0.44)	-
	Te_2	0.26 (0.18)	0.18 (0.12)	-
vl	Te_1	-	-	4.07 (3.54)
	Te_2	-	-	5.57 (7.71)
pvl	Te_1	0.23 (0.20)	0.90 (0.68)	4.72 (4.71)
	Te_2	0.26 (0.18)	0.87 (1.66)	6.25 (7.90)

In Fig. 7 the distributions of ε_p and ε_v are illustrated with box plots for the various labeling strategies on both test sets (Te_1 first and Te_2 second). For completeness, in Fig. 8 it is possible to see the same data generating cumulative performance plots (Te_1 solid and Te_2 dashed). From these figures, it is possible to better appreciate the same trends identified by the global metrics in Tab. III. In particular, the order of magnitude improvement in the performance when considering the LiDAR, the capability of the RNN to reconstruct the velocity, especially with data from the LiDAR, and finally the capability of the RNN to improve the position estimate, albeit only marginally, as it is possible to observe by comparing C_p or C_{pl} with R_p or R_{pl}. Finally, it is also noted that the training and validation envelopes chosen for the RNN demonstrated to have been chosen adequately for the testing conditions.

Finally, it is interesting to observe the reconstructed states of the CPOs of the Te_2 in the W reference frame both in the position and velocity phase spaces respectively in the scatter plots in Fig. 9 and Fig. 10. The size of the points in these plots represents either ε_p or ε_v , accordingly. In particular, in Fig. 9 an oscillation is visible mainly in the radial direction of the position estimate obtained only with optical observables. This is caused by the rotational state of the irregular shape model of Didymos. It is therefore concluded that the irregular shape model coupled with the rotation of the body is

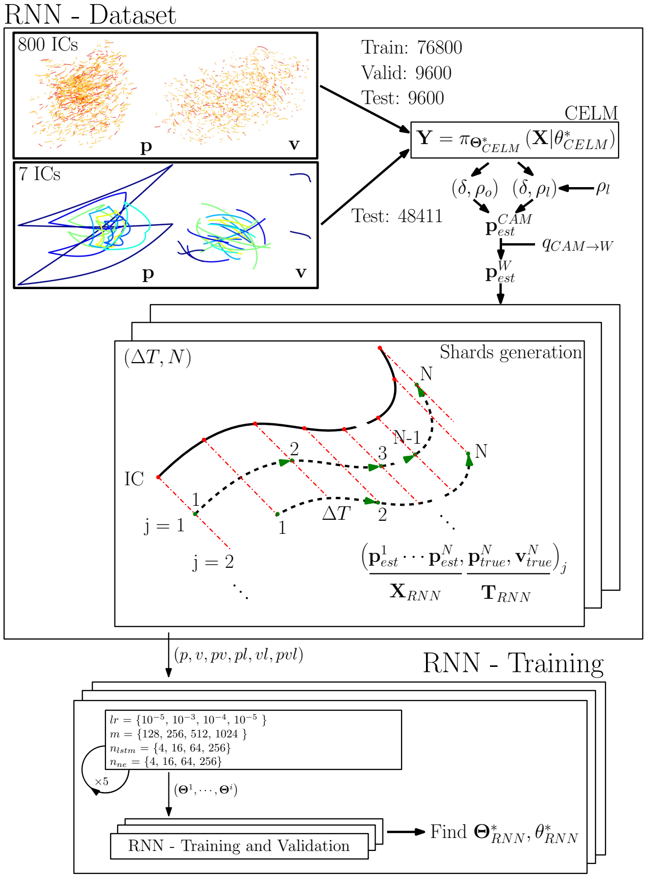

RNN - Dataset
800 ICs
p
v
7 ICs
p
v
Train: 76800
Valid: 9600
Test: 9600
Test: 48411
CELM
$\mathbf{Y} = \pi_{\Theta^*_{CELM}}(\mathbf{X}|\theta^*_{CELM})$
(δ, ρ_o)
(δ, ρ_l)
ρ_l
$\mathbf{p}^{CAM}_{est}$
$q_{CAM \to W}$
$\mathbf{p}^{W}_{est}$
$(\Delta T, N)$
Shards generation
IC
N
N-1
N
3
2
1
ΔT
2
1
j = 1
j = 2
$\underbrace{(\mathbf{p}^1_{est} \cdots \mathbf{p}^N_{est}}_{\mathbf{X}_{RNN}}, \underbrace{\mathbf{p}^N_{true}, \mathbf{v}^N_{true})_j}_{\mathbf{T}_{RNN}}$
(p, v, pv, pl, vl, pvl)
RNN - Training
$lr = \{10^{-5}, 10^{-3}, 10^{-4}, 10^{-5}\}$
$m = \{128, 256, 512, 1024\}$
$n_{lstm} = \{4, 16, 64, 256\}$
$n_{ne} = \{4, 16, 64, 256\}$
×5
$(\mathbf{\Theta}^1, \cdots, \mathbf{\Theta}^i)$
RNN - Training and Validation
Find $\mathbf{\Theta}^*_{RNN}, \theta^*_{RNN}$

Fig. 6. Sketch of the second portion of the training with the RNN.

introducing higher errors in the range estimate which are generated first in the C_p and then passed into R_p. These oscillations are clearly visible in the close-up view of the CPO trajectories projected in the XY plane of the W reference frame in Fig. 9. When the LiDAR is used, as in the R_{pl} and C_{pl}, the estimates are much closer to the true CPOs, and the oscillation phenomenon is not observed so strongly. This is an additional indicator this phenomenon is mainly induced by the errors in the radial direction with respect to Didymos, for which the LiDAR is capable to provide a better estimate.

Using the same visualization in the velocity phase space, Fig. 10 is obtained. First, it is possible to see that the estimates without the LiDAR (R_v) are much more loosely related to the true CPOs in velocity phase space, while the ones obtained with the LiDAR (R_{vl}) are indeed much more adherent to the true ones. Two interesting phenomena are also observed in Fig. 10. First, by comparing the training interval in Fig. 5 with the testing one of various CPOs, it is possible to see that CPO_1 is the only one that is not contained within the training envelope. Nonetheless, it is possible to see from Fig. 10, that they are reconstructed partially correctly in inference by the RNN, especially by R_{vl}. Second, of the two groups of trajectories in the velocity phase space of CPO_1 divided by the sign of V_y, it is possible to see that only the ones with positive V_y get reconstructed well in all 3 components. Interestingly, the ones with negative V_y are reconstructed well only in the X and Z components, as it is clearly visible from the XY view in Fig. 10. It is therefore concluded that a deficiency exists in the RNN architecture to generalize the prediction outside the training interval. However, this is limited by only a component and it is not clear in which other regions such behavior would occur. The choice to design CPO_1 outside of the velocity training envelope of the RNN has been made purposely to test the RNN generalization capabilities, which demonstrated to be an interesting phenomenon to be investigated.

From these results, a twofold conclusion can be drawn on the capability of the RNN to reconstruct position and velocity from sequences of multiple segmentation masks. First, it can be concluded that a RNN can be used to improve the reconstruction of the position vector. However, this improvement is negligible, especially when considering the case in which the LiDAR is used. Second, for what concerns the velocity, the RNN is not capable to generate an accurate estimate when only optical observables are used, which prompts the usage of the LiDAR to decrease the positioning error in the radial direction committed by the CELM. When optical observables are complemented by LiDAR ones, the RNN is capable to reconstruct the velocity accurately, and partially even outside the training envelope. Finally, when considering a mixed labeling strategy in which both the position and velocity are estimated by the RNN, this proved to be more complex than anticipated and generally brought slightly less accurate estimates than when considering the two of them singularly.

IV. CONCLUSIONS

In this paper, a set of RNN architectures is designed and trained with various sequences of position estimates obtained from optical observations of segmentation maps extracted by CELM and optionally complemented with LiDAR data.

It is proved that a RNN is capable of generating an accurate prediction of the position of a spacecraft around an asteroid from a sequence of position estimates. It is observed, however, that the improvement in the position accuracy is only marginal to a pinpoint image-position prediction with a CELM. It is also observed that a RNN can be successfully used to generate velocity estimates from sequences of position extracted from images. The estimate is influenced by the position reconstruction accuracy, which in turn is highly dependent on the error on the range, which being a value estimated across the boresight direction is notoriously more sensible in optical-based systems. To counteract this, the contribution of the rangefinder LiDAR for a better range prediction has been demonstrated to be of paramount importance both for a more accurate position estimate and most importantly for a more accurate velocity estimate. Finally, it is also observed that in all RNN architectures considered, a

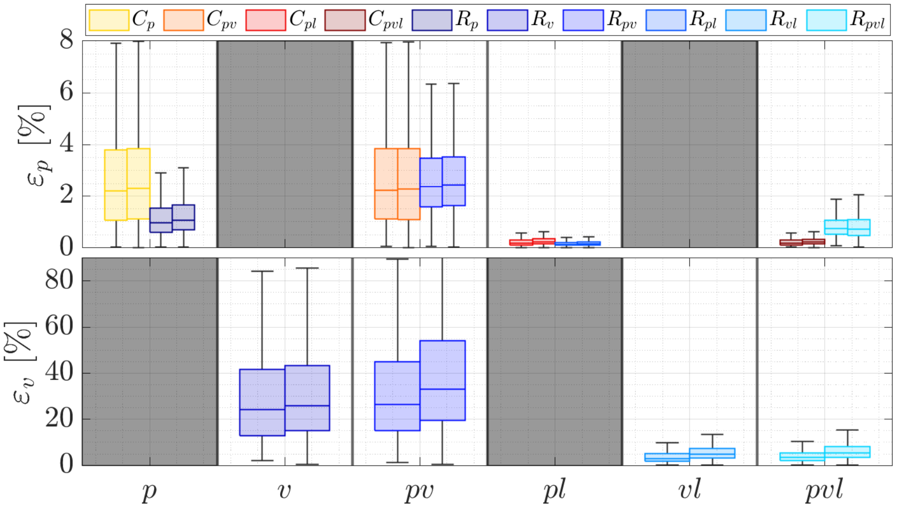


Fig. 7. Box plot of the position (top) and velocity (bottom) reconstruction error.

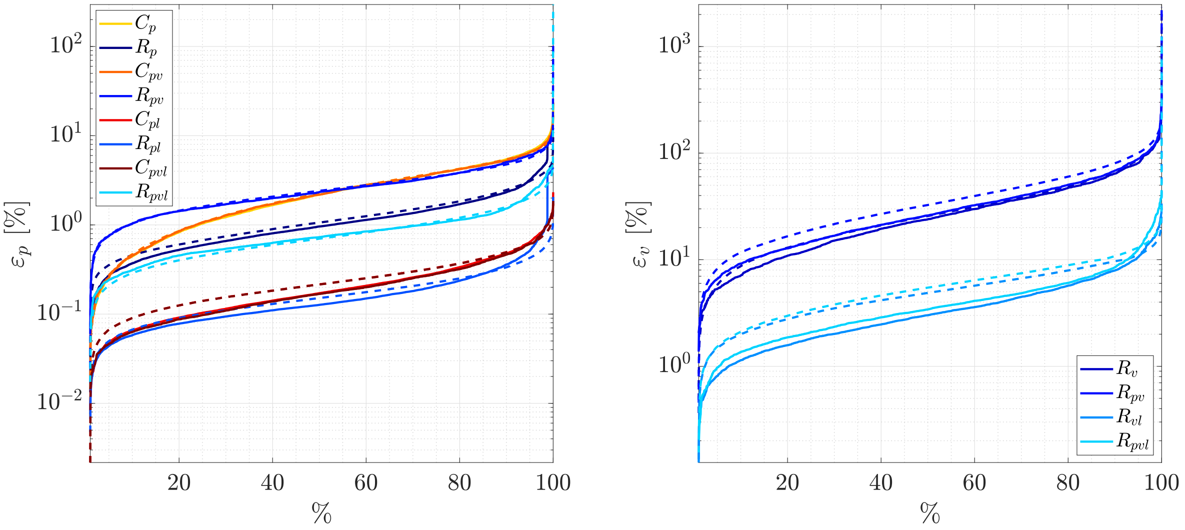


Fig. 8. Cumulative performance of the position (left) and velocity (right) reconstruction error.

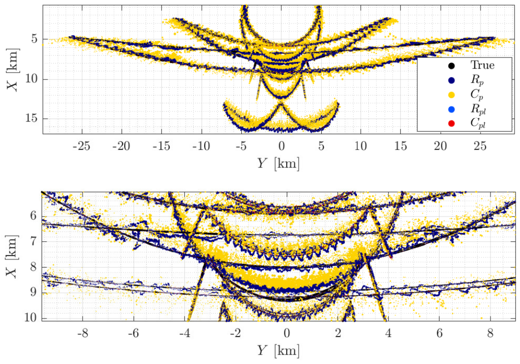


Fig. 9. True and estimated positions by C_p, C_{pl} and R_p, R_{pl}.

single 3-value position or velocity vector estimate has proven to be more simple and accurate to achieve than a combined 6-value position and velocity vector.

The RNNs described in this work offer the advantage over a traditional KF of not needing an a-priori initialization of the state, which could be of particular relevance for autonomous operations around an asteroid. On the other hand, the dynamic of the environment is embedded in the data used for training of the RNN. While an update of the dynamic in a KF could be implemented with a simple parameter change or by a different implementation of a given dynamical model, when considering the RNN approach presented in this work it would be necessary to generate new data reflecting the updated dynamics and retrain the RNN, which could be computationally intensive.

Future works will be focused on the inclusion of different dynamical settings to verify the generalization capability of the RNN and possible additional mechanisms to account for dynamic effects on the datasets. Only Didymos has been considered for this study, but the same methodology can be applied to any other small-body of interest by considering dedicated datasets. Also,

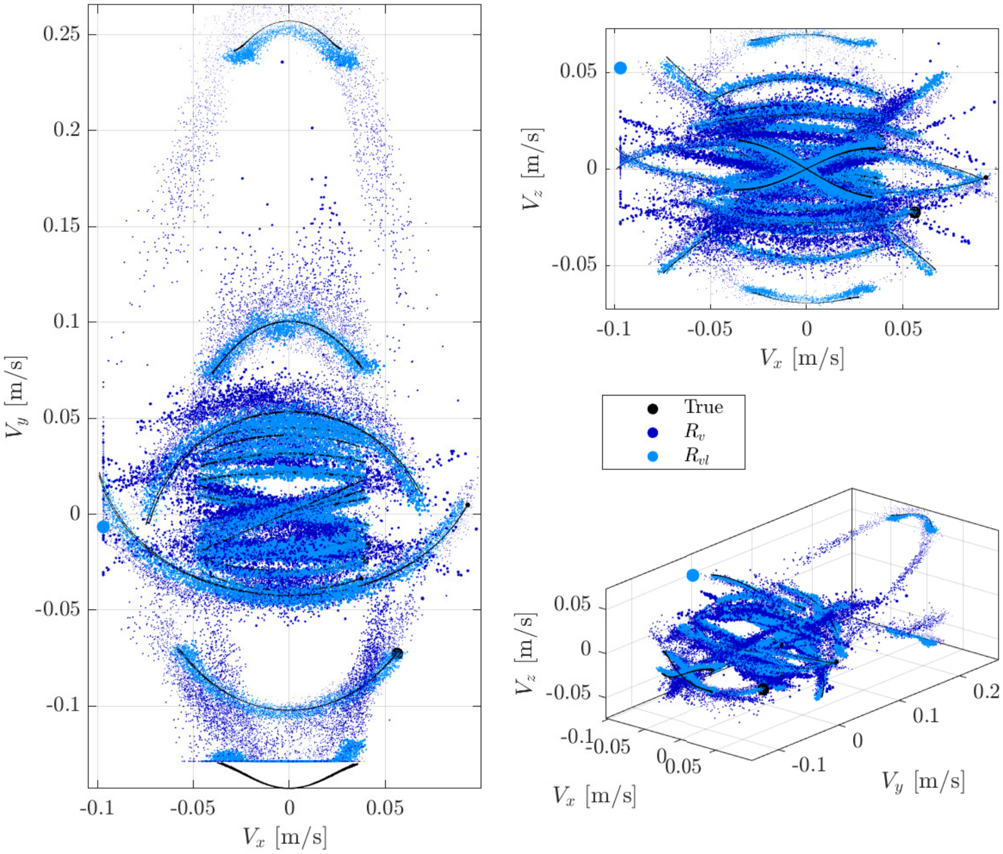


Fig. 10. True and estimated velocities by R_v and R_{vl}.

the LiDAR is assumed to be always available irrespective of the range from the asteroid, which is an assumption that may be invalid depending on the specific hardware considered. Including the simulation of a duty cycle of the LiDAR which could be activated only below a predetermined distance could be interesting to assess the effect on RNN, especially for what concerns the velocity. In the current baseline images are acquired at a constant frequency but the possibility to use variable time intervals could be investigated together with the addition of a time component in the labels.Finally, the segmentation maps considered as input in this work are ideal. The effect of real, onboard-generated maps as input is expected to cause a small drop in performance that could be interesting to investigate in future works.

APPENDIX

A. Details of the training of the Convolutional Extreme Learning Machine architectures

Each CELM architecture is designed with a hierarchical convolution structure, which seems to be the one better exploiting the random filtering capabilities [38]. While going deeper into the network towards the fully connected layer, the starting $128 \times 128 \times 1$ tensor is squeezed; its size is halved while its depth is doubled as a function of the depth level following a power law from 16 to 256. Each depth level is made up of the consecutive application of a convolution, activation function, and pooling operations, the convolutions being performed with 3×3 kernels. Having defined a procedural set of rules to generate each architecture, an ensemble of them is produced with the set of hyper-parameters $\boldsymbol{\Theta}$ illustrated in Tab. IV.

By considering the possible combination between d, K_d, A, and P a total of 120 different architectures are considered. To further extend the exploration, the random initialization of each kernel distribution is executed 5 times, producing a total of 600 architectures considered for training.

Once the forward pass of the CELM is executed on the entire dataset, for each architecture a regularized LS method is used with the 9 different regularization coefficients C illustrated in Tab. IV. Note that, as illustrated in [39], the training set is used to determine all possible values of $\boldsymbol{\beta}$ depending on C, while the validation set is used to determine the best value of C. For this purpose, the size of the validation set has been set arbitrarily to be a representative statistical sample. The combination of $\boldsymbol{\beta}$

Tab. IV

Sets of $\boldsymbol{\Theta}$ explored for the CELM architectures.

Symbol	Description	Possible values
d	Number of hidden layers in the convolutional architecture	1, 2, 3, 4, 5
K_d	Random distribution of the weight and biases of the kernels	Random Uniform (-1,1), Random Normal (0, 1), Orthogonal
A	Activation function used after the convolution operation	Normalized Rectified Linear Unit (nReLU), Rectified Linear Unit (ReLU), hyperbolic tangent (tanh), none
P	Pooling strategy after the activation function	Mean, Max
C	Coefficient of the regularized LS problem	10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}

determined from the training set and C from the validation set is used in inference on the test set to produce an estimate of the position.

The best architecture, $\pi_{\boldsymbol{\Theta}^*_{CELM}}$, is characterized by a 5 layers hierarchical pooling architecture which uses a *normalized ReLU* as activation function, a *mean* pooling strategy, and a *Random uniform* distribution of the kernels.

B. Details of the training of the Recurrent Neural Network architectures

Each RNN is generated in TensorFLow using a number of LSTM cells driven by n_{lstm} parameter. As activation function of the cells, an *hyperbolic tangent* is used while a *sigmoid* is used as recurrent activation function. The LSTM cells are followed up by a single layer made up of n_{ne} neurons which uses the *ReLU* as activation function. The hyper-parameter search serves the purpose to identify the values of n_{lstm}, n_{ne} , as well as m and lr , that characterize the best performing RNN architecture on the validation set. All the values of hyper-parameters tested during training of the RNN are illustrated in V.

Adopting the methodology illustrated in [42] (that recommends an iterative framework to find the optimal architecture design by an incremental exploration of the hyper-parameters space) with the empirical findings from [43] (that stresses the importance of gradual changes and global perspective while performing hyper-parameters tuning), a 5 steps iterative grid-search hyper-parameter finding is performed using *adam* optimizer with default settings and a *mean squared error* metric as the loss function for the components of the state vector.

In order to keep a consistent computational effort throughout different steps, at each iteration, the number of parameters to be tested is reduced while the epochs are increased. A tournament kind of training is thus performed as suggested from [42] defining as discriminative global metric the best mean position error and/or velocity error, using the ε_p and ε_v metrics defined in Section III in (3) and (4).

Tab. V

Sets of $\boldsymbol{\Theta}$ explored for RNN training.

Symbol	Description	Possible values
lr	Learning rate of *Adam* optimizer	10^{-3},10^{-4}, 10^{-5}
m	Batch size for gradient descent	128, 256, 512, 1024
n_{lstm}	Number of LSTM cells used in the RNN	4, 16,64, 256
n_{ne}	Number of neurons used in the RNN architecture	4, 16, 64, 256

The training is performed one labeling strategy at a time. In the first iteration, all possible combinations of the RNNs represented by the hyper-parameters Θ_{RNN} from Tab. V are trained and validated on the 5 different sets of shards in Tab. I for just 10 epochs, for a total of 1280 training episodes. The top best 128 sets of Θ_{RNN} which achieve the lowest global metric on the validation set are stored and passed on to the next iteration. In the second iteration, these 128 RNNs are trained from scratch for 50 epochs, out of which only 10 top-performing RNNs are passed into the next iteration. In the third iteration, these are trained for 250 epochs to generate the single best-performing set of Θ_{RNN}. This corresponds to only 1 out of the 256 possible ones illustrated in Tab. V. In the fourth iteration, this combination is further explored by 27 different combination in which m, n_{ne}, and n_{lstm} are changed around the best value found in the previous iteration to the closest one possible (note that the this corresponds to the next values following a power of 2 law, so that for example the closest values of $m = 64$ would be $m = 32$ and $m = 128$, which do not belong to the original values illustrated in Tab. I). The RNNs obtained with these settings are then trained for 250 epochs to generate a single setup. In the fifth and last iteration, this architecture is trained from scratch for 400 epochs by changing manually only the learning rate lr.

ACKNOWLEDGMENT

The authors would like to thank Claudio Bottiglieri and Fabio Ferrari for the support received in the design of the trajectory samples around the Didymos asteroid.

REFERENCES

[1] M. B. Quadrelli *et al.*
Guidance, navigation, and control technology assessment for future planetary science missions
Journal of Guidance, Control, and Dynamics, vol. 38, no. 7, pp. 1165–1186, Jul. 2015.

[2] J. Gil-Fernandez and G. Ortega-Hernando
Autonomous vision-based navigation for proximity operations around binary asteroids
CEAS Space Journal, vol. 10, no. 2, pp. 287–294, Feb. 2018. [Online]. Available: https://doi.org/10.1007/s12567-018-0197-5

[3] E. J. Lessac-Chenen *et al.*
Optical navigation operations and preparations for the lucy trojan-asteroid mission

In *AIAA SCITECH 2022 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2022. [Online]. Available: https://doi.org/10.2514/6.2022-1226

[4] D. Bekker, R. Smith, and M. Q. Tran
Guiding dart to impact — the fpga soc design of the draco image processing pipeline
In *2021 IEEE Space Computing Conference (SCC)*, 2021, pp. 122–133.

[5] C. Buonagura, M. Pugliatti, and F. Topputo
Image processing robustness assessment of small-body shapes
The Journal of the Astronautical Sciences, vol. 69, no. 6, pp. 1744–1765, Nov. 2022. [Online]. Available: https://doi.org/10.1007/s40295-022-00348-6

[6] S. Bhaskaran, J. E. Riedel, and S. P. Synnott
Autonomous nucleus tracking for comet/asteroid encounters: the stardust example
In *1998 IEEE Aerospace Conference Proceedings (Cat. No.98TH8339)*, vol. 2, 1998, pp. 353–365 vol.2.

[7] M. Pugliatti, V. Franzese, and F. Topputo
Data-driven image processing for onboard optical navigation around a binary asteroid
Journal of Spacecraft and Rockets, vol. 59, no. 3, pp. 943–959, May 2022. [Online]. Available: https://doi.org/10.2514/1.a35213

[8] J. A. Christian and S. B. Robinson
Noniterative horizon-based optical navigation by cholesky factorization
Journal of Guidance, Control, and Dynamics, vol. 39, no. 12, pp. 2757–2765, Dec. 2016. [Online]. Available: https://doi.org/10.2514/1.g000539

[9] A. Pellacani, M. Graziano, M. Fittock, J. Gil, and I. Carnelli
Hera vision based gnc and autonomy
. Proceedings of the 8th European Conference for Aeronautics and Space Sciences. Madrid, Spain, 1-4 july 2019, 2019. [Online]. Available: https://www.eucass.eu/doi/EUCASS2019-0039.pdf

[10] R. P. de Santayana and M. Lauer
Optical measurements for rosetta navigation near the comet
In *Proceedings of the 25th International Symposium on Space Flight Dynamics (ISSFD), Munich*, 2015.

[11] D. A. Lorenz *et al.*
Lessons learned from osiris-rex autonomous navigation using natural feature tracking
In *2017 IEEE Aerospace Conference*, 2017, pp. 1–12.

[12] N. Ogawa *et al.*
Image-based autonomous navigation of hayabusa2 using artificial landmarks: The design and brief in-flight results of the first landing on asteroid ryugu
Astrodynamics, vol. 4, no. 2, pp. 89–103, Jun. 2020. [Online]. Available: https://doi.org/10.1007/s42064-020-0070-0

[13] P. Michel *et al.*
The ESA hera mission: Detailed characterization of the DART impact outcome and of the binary asteroid (65803) didymos
The Planetary Science Journal, vol. 3, no. 7, p. 160, Jul. 2022. [Online]. Available: https://doi.org/10.3847/psj/ac6f52

[14] S. Henry and J. A. Christian
Absolute triangulation algorithms for space exploration
Journal of Guidance, Control, and Dynamics, vol. 46, no. 1, pp. 21–46, Jan. 2023. [Online]. Available: https://doi.org/10.2514/1.g006989

[15] E. Andreis, V. Franzese, and F. Topputo
Onboard orbit determination for deep-space CubeSats
Journal of Guidance, Control, and Dynamics, vol. 45, no. 8, pp. 1466–1480, Aug. 2022. [Online]. Available: https://doi.org/10.2514/1.g006294

[16] J. A. Christian, H. Derksen, and R. Watkins
Lunar crater identification in digital images
The Journal of the Astronautical Sciences, vol. 68, no. 4, pp. 1056–1144, Oct. 2021. [Online]. Available: https://doi.org/10.1007/s40295-021-00287-8

[17] R. Szeliski
Computer Vision, 2nd ed. Springer International Publishing, 2022. [Online]. Available: https://doi.org/10.1007/978-3-030-34372-9

[18] O. Ronneberger, P. Fischer, and T. Brox
U-net: Convolutional networks for biomedical image segmentation
2015.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla
Segnet: A deep convolutional encoder-decoder architecture for image segmentation
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 12, pp. 2481–2495, 2017.

[20] D. Thompson, S. Niekum, T. Smith, and D. Wettergreen
Automatic detection and classification of features of geologic interest
In *Proceedings. of IEEE Aerospace Conference*, 2005, pp. 366–377.

[21] K. L. Wagstaff, D. R. Thompson, B. D. Bue, and T. J. Fuchs
Autonomous Real-time Detection of Plumes and Jets from Moons and Comets
The Astrophysical Journal, vol. 794, no. 1, p. 43, Oct. 2014.

[22] T. J. Fuchs *et al.*
Enhanced flyby science with onboard computer vision: Tracking and surface feature detection at small bodies
Earth and Space Science, vol. 2, no. 10, pp. 417–434, Oct. 2015. [Online]. Available: https://doi.org/10.1002/2014ea000042

[23] A. Scorsoglio, A. D'Ambrosio, L. Ghilardi, B. Gaudet, F. Curti, and R. Furfaro
Image-based deep reinforcement meta-learning for autonomous lunar landing
Journal of Spacecraft and Rockets, vol. 59, no. 1, pp. 153–165, Jan. 2022. [Online]. Available: https://doi.org/10.2514/1.a35072

[24] K. Tomita, K. A. Skinner, and K. Ho
Bayesian deep learning for segmentation for autonomous safe planetary landing
Journal of Spacecraft and Rockets, vol. 59, no. 6, pp. 1800–1808, Nov. 2022. [Online]. Available: https://doi.org/10.2514/1.a35104

[25] T. Driver, K. Tomita, K. Ho, and P. Tsiotras
Deep monocular hazard detection for safe small body landing
2023. [Online]. Available: https://arxiv.org/abs/2301.13254

[26] E. Caroselli, F. Belien, A. Falke, F. Curti, and R. Forstner
Deep learning-based passive hazard detection for asteroid landing in unexplored environment
In *44th AAS GN&C conference, Colorado, Breckenridge*, no. AAS 22-044, Feb 2022, pp. 1–16.

[27] T. Claudet, K. Tomita, and K. Ho
Benchmark analysis of semantic segmentation algorithms for safe planetary landing site selection
IEEE Access, vol. 10, pp. 41 766–41 775, 2022. [Online]. Available: https://doi.org/10.1109/access.2022.3167763

[28] M. Pugliatti and M. Maestrini
Small-body segmentation based on morphological features with a u-shaped network architecture
Journal of Spacecraft and Rockets, vol. 59, no. 6, pp. 1821–1835, Nov. 2022. [Online]. Available: https://doi.org/10.2514/1.a35447

[29] J. Mukhoti and Y. Gal
Evaluating bayesian deep learning methods for semantic segmentation
2018, pre-print. [Online]. Available: http://arxiv.org/abs/1811.12709

[30] E. Hüllermeier and W. Waegeman
Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods
Machine Learning, vol. 110, no. 3, pp. 457–506, Mar. 2021. [Online]. Available: https://doi.org/10.1007/s10994-021-05946-3

[31] M. Pugliatti and F. Topputo
Navigation about irregular bodies through segmentation maps
In *31st Space Flight Mechanics Meeting, Charlotte, NC*, vol. 176. San Diego, CA: Univelt, Inc., Feb 2021, pp. 1169–1188.

[32] T. Weyand, I. Kostrikov, and J. Philbin
Planet-photo geolocation with convolutional neural networks

In *European Conference on Computer Vision*. Springer, 2016, pp. 37–55.

[33] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew
Extreme learning machine: Theory and applications
Neurocomputing, vol. 70, no. 1-3, pp. 489–501, Dec. 2006. [Online]. Available: https://doi.org/10.1016/j.neucom.2005.12.126

[34] G.-B. Huang
An insight into extreme learning machines: Random neurons, random features and kernels
Cognitive Computation, vol. 6, no. 3, pp. 376–390, Apr. 2014. [Online]. Available: https://doi.org/10.1007/s12559-014-9255-2

[35] G. Huang, G.-B. Huang, S. Song, and K. You
Trends in extreme learning machines: A review
Neural Networks, vol. 61, pp. 32–48, Jan. 2015. [Online]. Available: https://doi.org/10.1016/j.neunet.2014.10.001

[36] G.-B. Huang, Z. Bai, L. L. C. Kasun, and C. M. Vong
Local receptive fields based extreme learning machine
IEEE Computational Intelligence Magazine, vol. 10, no. 2, pp. 18–29, May 2015. [Online]. Available: https://doi.org/10.1109/mci.2015.2405316

[37] I. R. Rodrigues, S. R. da Silva Neto, J. Kelner, D. Sadok, and P. T. Endo
Convolutional extreme learning machines: A systematic review
Informatics, vol. 8, no. 2, p. 33, May 2021. [Online]. Available: https://doi.org/10.3390/informatics8020033

[38] A. M. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng
On random weights and unsupervised feature learning
In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, L. Getoor and T. Scheffer, Eds. Omnipress, 2011, pp. 1089–1096. [Online]. Available: https://icml.cc/2011/papers/551_icmlpaper.pdf

[39] M. Pugliatti and F. Topputo
Design of convolutional extreme learning machines for vision-based navigation around small bodies
2022. [Online]. Available: https://arxiv.org/abs/2210.16244

[40] F. Ferrari, V. Franzese, M. Pugliatti, C. Giordano, and F. Topputo
Trajectory options for hera's milani cubesat around (65803) didymos
The Journal of the Astronautical Sciences, vol. 68, p. 973–994, Sep. 2021.

[41] A. Accomazzo *et al.*
Rosetta operations at the comet
Acta Astronautica, vol. 115, pp. 434–441, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094576515002544

[42] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollar
Designing network design spaces
In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020, pp. 10 428–10 436.

[43] L. N. Smith
A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay
2018. [Online]. Available: https://arxiv.org/abs/1803.09820

Mattia Pugliatti received his B.Sc. degree in Aerospace engineering at Politecnico di Milano, Milan, Italy in 2015 and his M.Sc. in Space engineering at TU Delft, Delft, The Netherlands in 2018. During his studies he spent 4 months at Airbus Defense and Space, Friedrichshafen, Germany and 9 months at ISAS/JAXA, Sagamihara, Japan. After finishing his studies he worked for 1 year in GMV, Madrid, Spain as a GNC engineer on the phase B/B1 of the Hera mission. Since 2019 he is an Early Stage Researcher of the H2020 Stardust-R network and a Ph.D. student within the Deep-space Astrodynamics Research and Technology (DART) group, at Politecnico di Milano. During this period, he spent 4 months at the University of Arizona, Tucson, AZ. His research is focused on enhanced image processing for autonomous vision-based applications onboard CubeSats, especially around small-bodies.

Andrea Scorsoglio received his B.Sc. degree in aerospace engineering at Politecnico di Milano, Milan, Italy, in 2015, and his M.Sc. degree in space engineering at Politecnico di Milano, Milan, Italy, in 2018. Currently, he is working towards his Ph.D. in systems and industrial engineering at the University of Arizona, Tucson, USA. He is currently a Research Graduate Assistant at the Space Systems Engineering Laboratory at the University of Arizona. His research is focused on reinforcement learning and physics-informed neural networks applications to spacecraft guidance navigation and control, keplerian and non-keplerian orbital mechanics, and computer vision.

Roberto Furfaro is a Full Professor in the Department of Systems and Industrial Engineering, Department of Aerospace and Mechanical Engineering, University of Arizona. He is also the Deputy Director of the Space, Security, Safety & Sustainability Center (S4C). He obtained a Laurea Degree (M.S. equivalent) in Aeronautical Engineering (1998, University of Rome "La Sapienza") and a Ph.D. in Aerospace Engineering (2004, University of Arizona). He has served as PI and Co-PI of numerous high-impact research and development grants with a total amount of funds received by NASA, AFRL and other agencies over the past 15 years exceeding $80M. He published more than 85 peer-reviewed journal papers and more than 200 conference papers and abstracts. He is currently technical member of the AIAA Astrodynamics Committee and the AAS Space Surveillance Committee as well as Associate Editor for IEEE Transactions on Aerospace and Electronic Systems. During phase B-D (2011-2016) of the OSIRIS REx Asteroid Sample Return Mission, he was the systems engineering lead for the Science Processing and Operations (SPOC). He is currently the Target Follow-up WG lead for the NASA NEO Surveyor Mission. For his contribution to the OSIRIS REx mission, the asteroid 2003 WX3 was renamed 133474 Roberto Furfaro. Recently, Prof. Furfaro has been elected the 2021 Da Vinci Fellow at the College of Engineering, University of Arizona. Additionally, Prof. Furfaro has been elected AIAA Associate Fellow, Class 2022 and AAS Fellow, Class 2021.

Francesco Topputo is a Full Professor of Space Systems with Politecnico di Milano, Milan, Italy, and holds a position as Visiting Professor with TU Delft, Delft, The Netherlands. He is an ERC laureate (CoG 2019) and has been PI in 14 research projects. He leads the DART group, Politecnico di Milano. He has authored or co-authored 55 peer-reviewed articles in international journals and over 180 works in total. His research interests include spacecraft flight dynamics, interplanetary CubeSat mission and system design, autonomous guidance, navigation, and control.