THE IMAGE PROCESSING OF MILANI: CHALLENGES AFTER DART IMPACT

Mattia Pugliatti[(1)], **Carmine Giordano**[(1)], **Francesco Topputo**[(1)]

[(1)]*Politecnico di Milano, Via La Masa 34, 20156 Milano, mattia.pugliatti@polimi.it*

ABSTRACT

The Milani CubeSat mission aims to observe and study the Didymos binary system after deployment from the Hera mothercraft. The image processing of Milani uses optical observables of the primary body extracted from images to estimate its center of mass and enable autonomous onboard navigation. The algorithm employs a data-driven approach, with coefficients tuned based on observations and new data that can also be collected from other spacecraft. Using an updated version of the primary shape after the DART mission, a significant drop in performance has been observed due to the inability of the algorithm to establish a meaningful relationship between images and phase angles, challenging the original design of the image processing algorithm. In this work, this issue is investigated and proven to be caused by the oblate shape of the primary. Also, different alternative approaches are proposed considering additional parameters extracted from images, using polynomial chaos expansion, neural, and convolutional networks. The findings provide valuable insights for the adoption of data-driven methods in interplanetary missions, emphasizing the need for robust and adaptable algorithms to account for changes in target characteristics, especially when considering small-body missions.

1 INTRODUCTION

Milani is a 6U CubeSat that will be released in the Didymos environment by the Hera [15] mothercraft in early 2027. Milani will be released during a dedicated operational phase after an early characterization of the binary system. The Didymos system consists of a primary and a secondary body called Didymos (D1) and Dimorphos (D2), respectively. A peculiarity of the mission is that at the moment of the CubeSat release, the system will have already been observed closely at least twice: briefly by the DART [11] mission before impact on D2, and for a longer period by Milani's mothercraft Hera [15].

To accomplish its objectives, Milani is designed with both orbital and attitude control capabilities. Both the Guidance, Navigation, and Control (GNC) and Image Processing (IP) have been developed by the Deep-space Astrodynamics Research & Technology group (DART)[1] at Politecnico di Milano using Simulink 2020a[2] for its simplicity and the capability to convert high-level rapid prototyping code in Matlab/Simulink as C-code via auto-coding. This capability allows for fast iteration between the design of the algorithms and their integration with the onboard software.

Milani's onboard navigation strategy relies on optical observables of D1 extracted from images and then used in an on-board Extended Kalman Filter (EKF). To do so, a robust, simple, flexible, and accurate IP method is needed. D1 has been chosen as the main target of the IP since it is expected to be the largest, most visible, and regular body of the binary system.

[1]https://dart.polimi.it/, last accessed: 27th May 2023.

[2]https://www.mathworks.com/products/simulink.html, last accessed: 27th May 2023.

To account for the irregularities of D1 and to benefit from observations of the mission first by DART and then by Hera, the IP has been designed as a data-driven pipeline since the early stages of the project. Its architecture has been conceived to stay untouched from the end of development (Summer 2022) until the deployment within the binary system (Winter 2027). New observations about the properties of the system can then be used to tune the coefficients of the main data-driven functions used in different steps of the IP algorithm. To do so, new images are needed. These can be either synthetically generated or acquired from real sensors. A detailed description of the IP is out of the scope of this work but can be found in [18], [20]. Only a brief overview is given hereafter.
The IP can operate in three different modes, in order of increasing complexity these are called: COB, WCOB, and SSWCOB. In the first, the center of brightness (CoB) is used as an estimate of the center of mass (CoM) of D1, while in the latter two, a scattering correction is applied to the CoB coordinate to shift the estimated CoM of D1 towards the true CoM. This correction is critical when considering high-phase angles and the irregularity of a body such as in the case of asteroids and comets. The scattering correction is represented by the following equation:

$$\begin{bmatrix} CoF_x \\ CoF_y \end{bmatrix} = \begin{bmatrix} CoB_x \\ CoB_y \end{bmatrix} + \omega \cdot \mu(\Psi, \delta) \cdot \begin{bmatrix} \cos(\Phi) \\ \sin(\Phi) \end{bmatrix} \tag{1}$$

where Ψ is the phase angle, μ is the magnitude in pixels of the correction vector between CoB and CoM, Φ is an orientation function, and ω is a weight vector that can be used to tune the correction term. Both in the WCOB and SSWCOB, ϕ and μ are data-driven functions that depend on parameters extracted from a dataset of images. On the other hand, Ψ is a data-driven function only in the WCOB, while in the SSWCOB it is evaluated directly from data from the Sun sensor.
By design, the IP of Milani has been structured as a pipeline that uses coefficients to express these three data-driven functions. The coefficients can be changed throughout different phases of the Milani mission and through data obtained from other spacecraft. Major changes in the pipeline were thus not expected for the IP.
However, a preliminary assessment using an updated version of the shape model of D1 has proven deeply challenging for the current architecture. In this work, the authors address the nature of the issue, and its impact on the current pipeline, offering a solution that can be easily implemented on board, sharing relevant lessons learned regarding the adoption of data-driven IP methods for inter-planetary missions, and comparing together different alternatives.

2 CHALLENGES AFTER DART IMPACT

Prior to the arrival of DART, D1 was expected to be a top-shaped, mild irregular body with a volume equivalent diameter of 780 ± 30 m. Its principal axes were estimated to be $832\ m \pm 3\%$ m, $838\ m \pm 3\%$, and $786\ m \pm 5\%$ and were computed using radar observations [9], [10]. After DART, a new estimate of the shape of D1 returned a more oblate body with principal axes equal to $849\ m \pm 5.6\ m$, $851\ m \pm 5.6\ m$, and $620\ m \pm 5.6\ m$ [13], the last axes suffering the biggest variation. This change, especially across the z-axis, has been observed to cause a major drop in performance of the WCOB mode of the IP of Milani. On the other hand, other changes in the model, such as the distribution of boulders over the surface, did not play a role, since the IP only uses global and not local properties of D1.
Interestingly, the drop in performance has been observed to be caused by the data-driven function Ψ used to estimate the phase angle from the image. In its original design, a useful relationship could be established between the eccentricity of the ellipse fitting the blob of pixels associated to D1 with the same normalized second central moment and the phase angle Ψ.

To visualize this relationship for different values of scaling across the z-axis s_z, datasets made of 5000 images each, randomly generated in a spherical shell about D1 with an excursion of Ψ up to 120 deg, are used. For the purpose of this study, images are rendered in Blender using the *Cycles* rendering engine at 1024×1024 pixels with a field of view of 5.5 deg. The images are also rendered with an ideal pointing towards the CoM of D1 and with a random boresight rotation. Domain randomization is applied by adding a variable artificial noise to each image and by changing the albedo and coefficients of the scattering law used at each acquisition.
When considering the original scale of D1 from [9] ($s_z = s_0$), a second-order polynomial has been chosen to represent the Ψ-e relationship, as illustrated in Figure 1. Such a relationship was also tested for a variation of the scale across the z-axis up to ± 5 % of the original value s_0 [16] and proved to cause minor fluctuations in the performance of the IP.

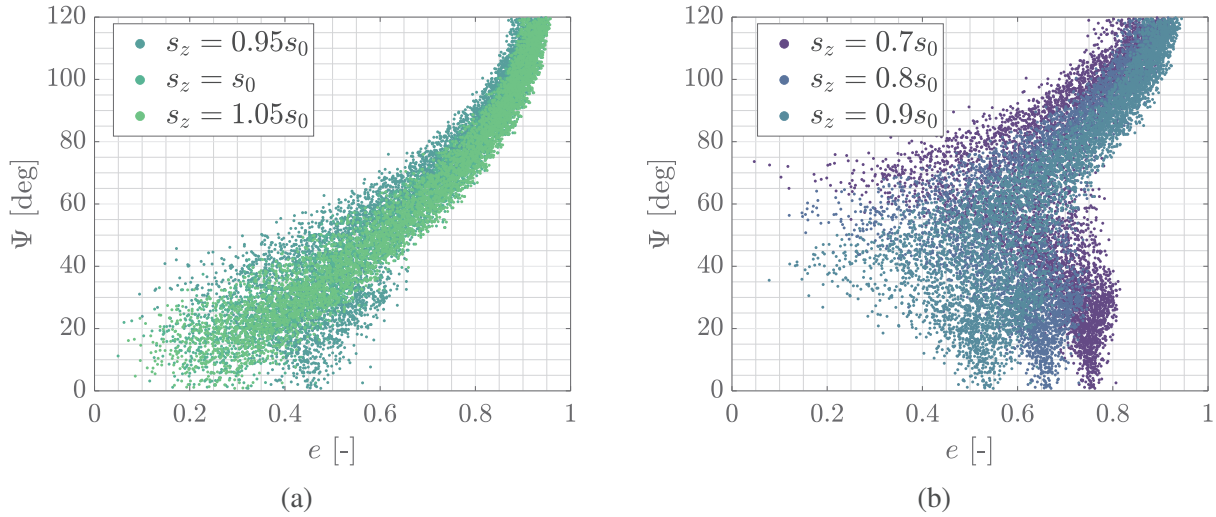
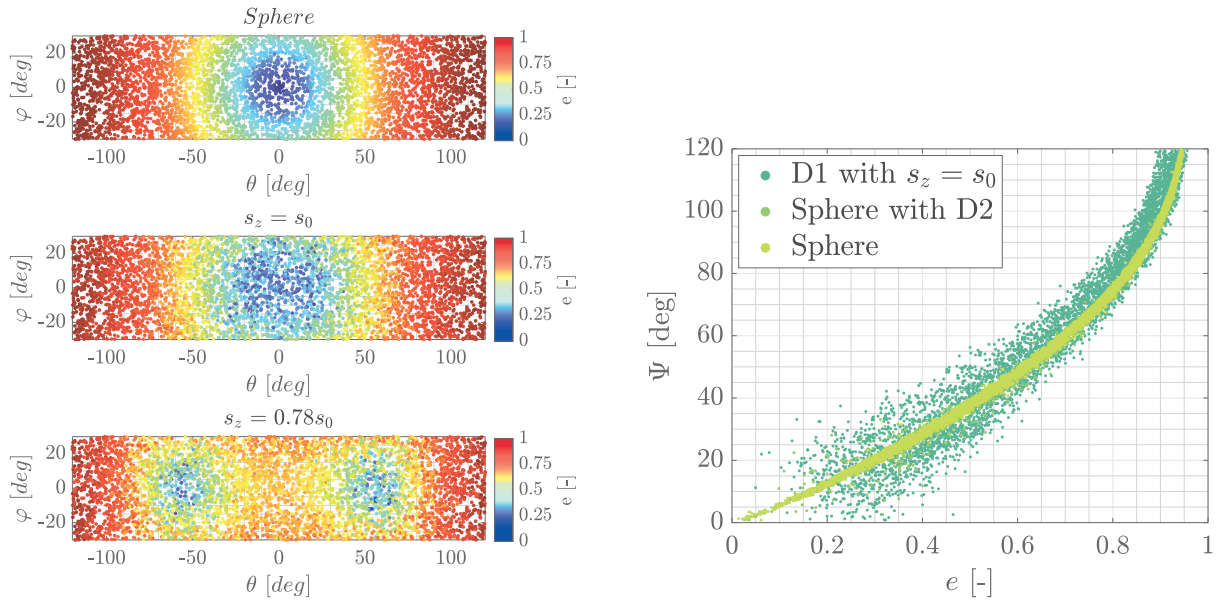


Figure 1: Phase angle Ψ as a function of the eccentricity of the blob of pixels associated with D1 for different values of scaling across the z-axis s_z for 5000 images randomly distributed about D1.s_0 represent the scale of D1 before the DART mission update.

However, as it is possible to see in Figure 1 when considering higher values of oblateness, a clear functional relationship cannot be established anymore considering the eccentricity as the sole parameter. This is caused by a specific interaction between the ellipse fit and oblate objects.
As it is possible to see in Figure 2 by visualizing e as a function of polar angles around D1, increasing the oblateness of the body passing from a sphere, to D1 with $s_z = s_0$, to D1 with $s_z = 0.78 s_0$, the eccentricity passes from having one minimum to multiple ones. When considering a spherical object, the eccentricity of the fitted ellipse monotonically increases with the phase angle, as it is possible to see from Figure 3. This allows to establish a clear and useful relationship. On the other hand, when considering a highly oblate object, the minimum eccentricity is not associated anymore with the projection of the object at low phase angles. Given the irregularity of the object, a considerable offset may be present and the minimum eccentricity could be achieved with a proper combination between phase angle and view of the object that does not occur at low phase angles. This phenomenon creates the bifurcation visible both in Figure 1 and Figure 2.
It is also noted here that both the local irregularities over D1 shape and the presence of D2 in the images act purely as disturbances. This is also clearly visible from Figure 2, which illustrates what happens when a spherical body is substituted to D1 and subsequently when D2 is removed from the rendering software.
In conclusion, when considering more oblate shapes for D1, the eccentricity alone cannot be used as a parameter to predict the phase angle Ψ directly from images. In the next section, the authors provide



(a) (b)

Figure 2: (a) Eccentricity of the blob of pixels associated to D1 in polar coordinates for three different cases. (b) Phase angle Ψ as a function of the eccentricity of the blob of pixels associated with D1 with $s_z = s_0$ compared with a spherical body with and without D2.

alternative approaches to perform such an estimate while considering additional parameters that can be extracted from images that may be more suitable for the given task.

3 PROPOSED APPROACHES

The dataset of 5000 images in which D1 has $s_z = 0.78s_0$ is passed through the IP to generate a set of additional variables that could be extracted onboard. The purpose of this activity is to understand which additional variables that can currently be extracted onboard can be used to establish a meaningful relationship with the phase angle and recover performance. Each image is thus associated with a feature vector composed of 14 components, divided into three main groups: associated directly with the blob of pixels of D1 (f_{D1}), associated with the blob of pixels of the edge region of D1 (f_{edge}), and combined properties between f_{D1} and f_{edge} (f_{comb}). Finally, for each image, a feature vector of 14 elements is composed as $f_x = [f_{D1}, f_{edge}, f_{comb}]$.
To compose f_{D1} and f_{edge} the area (ν_{area}), perimeter (ν_{per}), circularity (ν_{circ}), extent (ν_{ext}), and eccentricity (ν_e) are put together as follows:

$$f_{D1} = [\log_{10}(\nu_{area}), \log_{10}(\nu_{per}), \nu_{circ}, \nu_{ext}, \nu_e]^{D1} \tag{2}$$

$$f_{edge} = [\log_{10}(\nu_{area}), \log_{10}(\nu_{per}), \nu_{circ}, \nu_{ext}, \nu_e]^{edge} \tag{3}$$

On the other hand, to compose f_{comb} more complex relationships are used based on previous experience. ν_1 is evaluated as the ratio between the perimeter of the blob of pixels of D1 and the sum of the perimeters of the multiple edge regions detected in the image. ν_2 is evaluated as the ratio between the perimeter of the blob of pixels of D1 and the sum of the areas of the multiple edge regions detected in the image. ν_3 is computed as a summation over the entire image of the normalized activation map after the application of a Sobel filter to the image. Finally, ν_4 is computed as the ratio between the

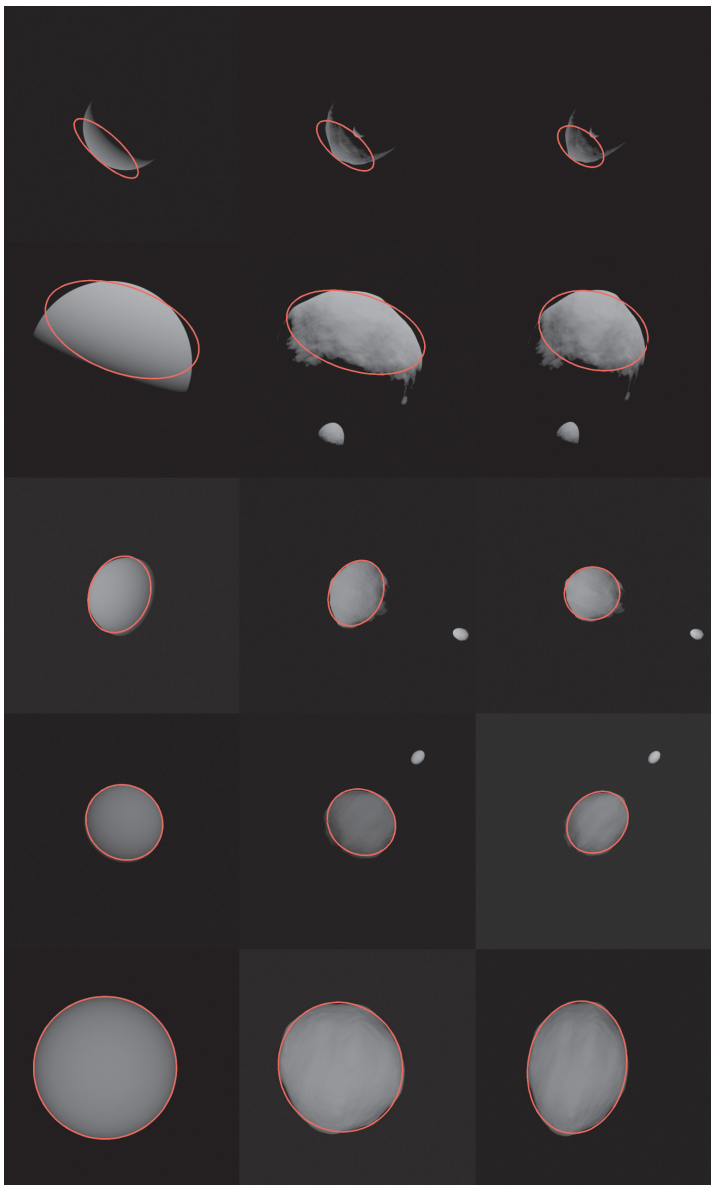


Figure 3: Mosaic of different views of a sphere (left), D1 with $s_z = s_0$ (center), and $s_z = 0.78 s_0$ (right) at different equatorial angles θ (from top to bottom 120, 90, 60,30, and 0 deg). The red curve represents the fitting ellipse with the same normalized second-order moment.

eccentricity of the blob of pixels of D1 and the eccentricity of the edge region. These components are then put together as:

$$f_{comb} = [\tanh(\nu_1), \tanh(\nu_2), \log_{10}(\nu_3), \nu_4]^{comb} \tag{4}$$

In the next section, five different alternatives are designed using either the dataset made of features f_x or directly the images. To allow a fair comparison between the different methods, all are allowed to use 4000 samples for training, 500 samples for validation (if necessary), and 500 samples for testing.

3.1 Polynomial chaos expansion

3.1.1 Introduction to Polynomial Chaos Expansion

Polynomial Chaos Expansion (PCE) is a nonlinear uncertainty quantification technique, introduced by Wiener in 1938 for the fluid dynamics domain [1], and then expanded to be used in different fields to both estimate the impact of uncertainties on some quantities of interest (QoI) and evaluate the evolution in time of an uncertain state. The main advantages of PCE are related to its ability to provide high-order accurate estimation of the statistics of the QoI, while requiring significantly less computational effort of other similar methods, such as Monte Carlo simulations [8]. In PCE, the solution is approximated using a series expansion based on some orthogonal polynomials. Within this framework, quantities of interest $\hat{\mathbf{x}}$ can be written as [3]

$$\hat{\mathbf{x}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{\alpha} \in \Lambda_{p,d}} \mathbf{c}_{\boldsymbol{\alpha}} \psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}) \tag{5}$$

where $\Lambda_{p,d}$ is a set of the multi-index of size d and order p defined on nonnegative integers, $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_d]$ is the set of input random variables, in which each element ξ_i is an independent identically distributed variable. The basis functions $\{\psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})\}$ are multidimensional spectral polynomials, orthonormal with respect to the joint probability measure $\rho(\boldsymbol{\xi})$ of the vector $\boldsymbol{\xi}$

$$\int_{\Gamma^d} \psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}) \psi_{\boldsymbol{\beta}}(\boldsymbol{\xi}) \rho(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\xi} = \delta_{\boldsymbol{\alpha}\boldsymbol{\beta}} \tag{6}$$

with Γ^d representing the d-dimensional hypercube where the random variable $\boldsymbol{\xi}$ are defined and $\delta_{\boldsymbol{\alpha}\boldsymbol{\beta}}$ is the Kronecker delta function. Thus, the basis functions choice depends only on $\rho(\boldsymbol{\xi})$. For instance, Hermite polynomials are the basis for normal random variables, while Legendre orthogonal polynomials are bases for the uniform distribution [1].
Generation of a PCE series means computing the generalized Fourier coefficients $\mathbf{c}_{\boldsymbol{\alpha}}$ by projection of the exact solution $\mathbf{x}(\boldsymbol{\xi})$ onto each basis function $\psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})$, truncated at the total order p

$$\mathbf{c}_{\boldsymbol{\alpha}} = E\left[\mathbf{x}(\cdot)\psi_{\boldsymbol{\alpha}}(\cdot)\right] = \int_{\Gamma^d} \mathbf{x}(\boldsymbol{\xi}) \psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}) \rho(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\xi} \tag{7}$$

PCE coefficients can be estimated by performing a Galerkin projection of the governing stochastic equations onto the $\{\psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi})\}$ subspace (the so-called, *intrusive* method), or solving a least-square regression or pseudospectral collocation (in the so-called, *non-intrusive* methods) [6]. The pseudospectral collocation solves directly the stochastic integral in Eq. (7) and thus it requires a straightforward function that links the input uncertainty with the QoI. In case of implicit or hidden connection between inputs and outputs, the least-square regression is the most suitable approach, since it does not require the outputs evaluated in specific collocation points.
The least-square regression is based on a random sampling of the uncertainties, that are then used to compute a pool of quantities of interests. The PCE coefficients are computed starting from this synthesized samples $\mathbf{x}(\boldsymbol{\xi})$, imposing that the sum of their difference with respect to the PCE expansion $\hat{\mathbf{x}}(\boldsymbol{\xi})$, at the sample points $\boldsymbol{\xi}_i$ is minimized, that is [6]

$$\mathbf{c}_{\boldsymbol{\alpha}} \simeq \arg\min_{\{\tilde{\mathbf{c}}_{\boldsymbol{\alpha}}\}} \frac{1}{M} \sum_{i=1}^{M} \left(\mathbf{x}(\boldsymbol{\xi}_i) - \sum_{\boldsymbol{\alpha} \in \Lambda_{p,d}} \tilde{\mathbf{c}}_{\boldsymbol{\alpha}} \psi_{\boldsymbol{\alpha}}(\boldsymbol{\xi}_i) \right)^2 \tag{8}$$

where M is the number of considered samples. The solution to this problem can be conveniently written in matrix form as

$$\left(\boldsymbol{\Psi}^T \boldsymbol{\Psi}\right) \mathbf{c} = \boldsymbol{\Psi}^T \mathbf{X} \tag{9}$$

where $\mathbf{c} \in \mathbb{R}^{P\times n}$ is the least-squares approximation of the PC coefficients, $\mathbf{X} \in \mathbb{R}^{M\times n}$ is the matrix containing the realizations of the vector $\mathbf{x}(\boldsymbol{\xi})$ and $\boldsymbol{\Psi} \in \mathbb{R}^{M\times P}$ is the measurement matrix, defined as

$$\boldsymbol{\Psi}[i,j] = \psi_{\alpha_j}(\boldsymbol{\xi}_i) \qquad \text{with} \quad i = 1,\dots,M, \quad j = 1,\dots,L \tag{10}$$

The number of samples M needed is not straightforward and it depends basically on the uncertainties, the polynomial basis, and the required accuracy.

3.1.2 The Arbitrary Polynomial Chaos

PCE requires the inputs to have a prescribed distribution, that must also enjoy the existence of a set of associated spectral orthogonal polynomials. However, natural phenomena and uncertainty in engineering cannot be always described using predefined known probability distributions. For this reason, the arbitrary Polynomial Chaos (aPC) technique has been introduced [5]. Under this framework, the basis functions are not drawn from the Wiener-Askey polynomials according to the input distribution [1], but they are constructed starting from the input data values, that can be provided in any form. Defining the polynomial of degree k in the (1-dimensional) random variable ξ

$$\psi_k(\xi) = \sum_{l=0}^{k} a_l^{(k)} \xi^l, \qquad \text{with} \quad k = 0,\dots,p \tag{11}$$

the problem associated with the arbitrary Polynomial Chaos (aPC) is to determine the polynomial coefficients $a_l^{(k)}$ for each ψ_k, such that the polynomial forms an orthonormal basis for the input arbitrary distribution. Starting from Eq. (6), the condition of orthogonality for each polynomial ψ_k with all the lower order polynomial, from 0 to $k-1$, can be written as

$$\int_\Gamma \left(\sum_{l=0}^{j} a_l^{(j)} \xi^l\right)\left(\sum_{l=0}^{k} a_l^{(k)} \xi^l\right) \mathrm{d}\xi = 0 \qquad \text{with} \quad j = 0,\dots,k-1 \tag{12}$$

The additional condition

$$a_k^{(k)} = 1$$

that acts as an intermediate surrogate for the normality condition, is added to Eq. (12) in order to close the system.
Recalling that the l-th order moment of the random variable ξ is

$$\mu_l = \int_\Gamma \xi^l \,\mathrm{d}\xi$$

the system in Eq. (12) can be rewritten in matrix form for each k as [5]

$$\begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_{k-1} & \mu_k \\ \mu_1 & \mu_2 & \cdots & \mu_k & \mu_{k+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu_{k-1} & \mu_k & \cdots & \mu_{2k-2} & \mu_{2k-1} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0^{(k)} \\ a_1^{(k)} \\ \vdots \\ a_{k-1}^{(k)} \\ a_k^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{13}$$

Thus, it is possible to compute the orthogonal basis starting only from the moments, without the need to determine the full distribution function. Moreover, the basis can be determined only if moments up to order $2d-1$ exist. In case the input is provided as a set of points, this condition is equivalent to

having at least k distinct values.
Once the coefficients have been determined, the orthogonal basis should be also normalized, by evaluating its norm as

$$\|\psi_k(\xi)\| = \left(\int_\Gamma \psi_k^2(\xi)\ \mathrm{d}\xi\right)^{\frac{1}{2}} \tag{14}$$

and redefining all the polynomial coefficients as

$$a_l^{(k)} = \frac{a_l^{(k)}}{\|\psi_k(\xi)\|}$$

These polynomials can be then used as the basis for PCE series since it has been proven they enjoy the same properties of convergence and accuracy [5].

3.1.3 Polynomial Chaos and Milani IP

PCE, and aPC, have been introduced and exploited for uncertainty quantification. However, recently PCE has been proven to be an effective technique also in other fields, e.g., it has been used to successfully propagate all-in-once a bundle of trajectories in a deterministic setting [14]. Starting from this point, a wider use for PCE in data-driven approaches can be devised since it shows some useful properties. As a matter of fact, PCE (and aPC) can be used as an effective interpolation method, not requiring to define a-priori the interpolant functions but selecting them automatically starting from the input samples, so that they possess spectral convergence with respect to the input variables. Furthermore, the same input samples can be used to find the generalized Fourier coefficients exploiting the least-squares approximation technique.
These desirable properties can be exploited to overcome the issue characterized in the previous section caused by D1 oblateness by using the feature vector f_x for every image as input ξ for PCE, in order to estimate the phase angle of Didymos. Since it is not guaranteed that they are statistically independent, Principal Components Analysis (PCA) is exploited to remove correlation among the different features. PCA output is then fed to aPC to build the orthogonal basis since feature distribution is not known a-priori. Thus, the procedure to estimate the phase angle of D1 for Milani, exploiting PCE and labeled *PCE-Full*, can be summarized as:

1. All the features extracted from the images f_x are used as inputs for the PCA to find the principal components coefficients C, and the explained variance s;
2. Principal components are computed as $\boldsymbol{\xi} = Cf_x$ and sorted by their explained variance;
3. The first dth variables are used to estimate the statistical raw moments and in turn the polynomial basis coefficients $a_l^{(k)}$ (as per Eq. (13)), and the basis polynomials (Eq. (12));
4. The principal components associated to the first $M = 4000$ images are used to compute the PCE coefficients $\mathbf{c}_{\boldsymbol{\alpha}}$ (Eq. (9));
5. The performance of the algorithm is evaluated by comparing the values predicted by the PCE (using Eq. (5)) with respect to the exact phase angle of the last $N = 500$ samples.

An alternative methodology, labeled *PCE-Res*, can be devised to perform the PCA only on the first dth features, following the sorting order given by point 2. This approach can help the IP when performed on-board since it reduces the number of features that should be extracted from each image.

3.2 Neural network

A Hyper-parameter search is performed varying the number of hidden layers, the number of neurons, and the activation function of a simple feed-forward Neural Network (NN) using the Matlab regression learner application. The first and last layers of the network are made respectively of 14 and 1 neurons. During training, PCA is enabled on the input variables. As the output of the hyper-parameter search, the best architecture is found to be made of 3 hidden layers, made respectively of 124, 8, and 46 neurons, all using ReLU as an activation function.
In this work, the NN is used to investigate whether or not a network can yield better results than PCE as a universal interpolation function.

3.3 Convolutional networks

An approach is also designed that makes use of Convolutional Extreme Learning Machine (CELM) and Convolutional Neural Network (CNN) that is working directly over the images, bypassing the feature vector f_x. The purpose of this approach is to use the best performing CNN to determine whether or not a better estimate can be performed directly from the images. This could provide a hint about the completeness of the 14 features selected to represent each image and whether additional information can be used from the images to estimate Ψ that is not currently explicitly encoded by the feature vector.
The training strategy used in this work is the same as the one illustrated in [19] while the images are preprocessed with the same algorithm illustrated in [17]. Both are only briefly described in this work. The training of the networks is performed using a Tesla P100-PCIE 16Gb GPU, with a 27.3 Gb of RAM in Google colab[3].
Following the same procedure illustrated in [19], 540 CELM hierarchically organized convolutional pooling architectures are generated. These alternate sequences of convolutions, activation functions, and pooling until the generation of a latent feature vector [7]. Training happens by flowing forward the dataset on the architectures to generate such vectors which are then linked with the target output via weighted connections β, as illustrated in Figure 4.

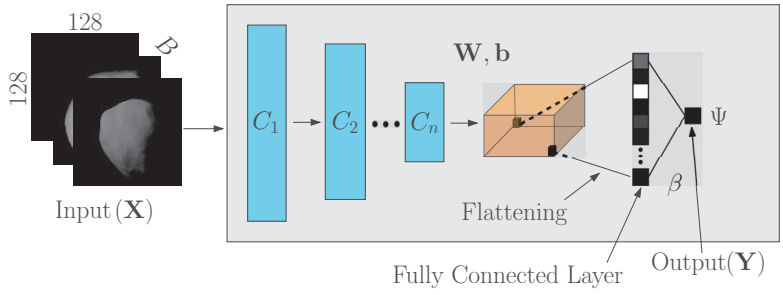


Figure 4: Visual representation of the architecture of the convolutional networks used in this work.

In the CELMs, the weights and biases of the network are set randomly at initialization, while the weights between the latent vector and the output layer are found via a regularized least-square method [7], [12]. The advantage of using CELMs in this stage is motivated by the very fast training time, which allows for efficient exploration of the architecture design space [4]. A thorough architecture design search involves the weights and bias initialization strategy (Random, uniform, orthogonal), the type of activation function to be used (none, ReLU, leaky ReLU, tanh, sigmoid), the pooling strategy to be used (mean or max), the number of sequences of convolution, activation function, and pooling

[3]https://colab.research.google.com/, retrieved 27 of May, 2023.

to be used in the architecture (from 2 to 6), and the values of the regularization parameter of the least-square problem (from 0.0001 to 10000 in increasing order of magnitudes).
After finding the optimal architecture setup, the CNN is trained at varying values of batch size (B), learning rate, and epochs. The best performing architecture is then selected as the one represented by the weights and biases at the minimum value of mean squared error on the validation set. The architecture of the CNN (which up to the fully connected layer *FC6* is the same as the CELM) is summarized in Figure 1 using TensorFlow 2.10 notation.

Table 1: Architecture of the 6 layers CNN. The *max* pooling strategy and the *normalized ReLU* are used. The networks has a total count of 2,884,737 parameters.

Layer acronym	Layer type	Output	Number of parameters
I	(InputLayer)	(None, 128, 128, 1)	0
C1	(Conv2D)	(None, 128, 128, 16)	160
A1	(nReLU)	(None, 128, 128, 16)	0
P1	(MaxPooling2D)	(None, 64, 64, 16)	0
C2	(Conv2D)	(None, 64, 64, 32)	4640
A2	(nReLU)	(None, 64, 64, 32)	0
P2	(MaxPooling2D)	(None, 32, 32, 32)	0
C3	(Conv2D)	(None, 32, 32, 64)	18496
A3	(nReLU)	(None, 32, 32, 64)	0
P3	(MaxPooling2D)	(None, 16, 16, 64)	0
C4	(Conv2D)	(None, 16, 16, 128)	73856
A4	(nReLU)	(None, 16, 16, 128)	0
P4	(MaxPooling2D)	(None, 8, 8, 128)	0
C5	(Conv2D)	(None, 8, 8, 256)	295168
A5	(nReLU)	(None, 8, 8, 256)	0
P5	(MaxPooling2D)	(None, 4, 4, 256)	0
C6	(Conv2D)	(None, 4, 4, 512)	1180160
A6	(nReLU)	(None, 4, 4, 512)	0
P6	(MaxPooling2D)	(None, 2, 2, 512)	0
FC6	(Flatten)	(None, 2048)	0
D1	(Dense)	(None, 512)	1049088
DO1	(Dropout)	(None, 512)	0
D2	(Dense)	(None, 512)	262656
DO2	(Dropout)	(None, 512)	0
D3	(Dense)	(None, 1)	513

4 RESULTS

From the PCA analysis, the components of the feature vector f_x are ordered in decreasing role of importance according to the RRelieF metric [2] as illustrated in Table 2. It is noted that the eccentricity of the blob of pixels associated with D1 is still the most important feature, as in the original relationship used in the WCOB of the IP of Milani.
Considering the PCE approach, the number of variables considered in the results varies from 1 to 12, in order to establish how many features are needed as inputs. It is important to note that even if the number of features is 14, the last 2 are ignored since they describe less than 0.01% of the variance.
Figure 5 shows the error histograms between the true and predicted phase angle of the PCE at varying numbers of inputs considered following the order in Table 2 for the PCE-Full and PCE-Res methods.

Table 2: Otput of the PCA: features of f_x ordered in descending order using the RRelieF metric.

Feature	RRelieF
ν_e^{D1}	1.8487e-03
ν_{ext}^{D1}	1.5718e-03
$\log_{10}(\nu_3)$	1.3916e-03
ν_4	1.1676e-03
$\log_{10}(\nu_{area}^{edge})$	9.5524e-04
ν_{circ}^{D1}	8.6640e-04
ν_e^{edge}	6.0699e-04
$\log_{10}(\nu_{per}^{edge})$	5.4506e-04
ν_{ext}^{edge}	2.3647e-04
$\tanh(\nu_1)$	7.1865e-05
$\tanh(\nu_2)$	-2.9021e-05
ν_{circ}^{edge}	-2.3132e-04
$\log_{10}(\nu_{area}^{D1})$	-3.6348e-04
$\log_{10}(\nu_{per}^{D1})$	-1.0600e-03

Table 3: Performance comparison between the methods considered in this work for the estimation of Ψ.

Value	WCOB	PCE-Full	PCE-Res	NN	CELM	CNN
μ [deg]	8.349	0.666	0.580	0.899	-1.453	0.854
σ [deg]	25.026	9.168	9.560	8.472	17.145	4.832
Q_{67} [deg]	24.963	3.447	3.863	3.169	6.310	2.690
Q_{95} [deg]	35.029	16.651	16.869	15.433	26.161	8.724

Another visualization of the same error is illustrated in Figure 6 which lists the values of σ and Q_{95} for both the PCE-based methods. Considering up to 9 input features gives the minimum value of variance for both methods. This value is thus considered for the comparison of the PCE-based methods with NN, CELM, and CNN ones.
Considering the same test set of 500 samples for each method, the error is computed as $\varepsilon = \Psi_{true} - \Psi_{pred}$. In Figure 7 is possible to see a visual comparison of the histogram errors of the different methods. Also, the performance of the WCOB is visualized out of interest and as representative of the original formulation of the IP of Milani considering only ν_e^{D1} as a parameter to estimate the phase angle. The performance of all methods is also summarized in Table 3, reporting for each one the mean, variance, Q_{67}, and Q_{95} values. Considering both Figure 7 and Table 3 the alternatives considered can be divided into three main groups. The first one is represented by the original version of the IP of Milani (WCOB) and the CELM methods, both performing very badly. This was expected from both cases. The second group is represented by the PCE-based methods and the NN. Both achieve similar metrics, also similar to the previous performance of the original IP of Milani when considering $s_z = s_0$ (for reference, $\mu = -6.114$ deg, $\sigma = 7.365$ deg, $Q_{67} = -3.117$ deg, and $Q_{95} = 6.319$ deg). Finally, the third group is represented by the CNN, that outperforms all previous methods considered.

5 CONCLUSIONS

In this work, the authors have addressed an issue caused by an unforeseen major update over the oblateness of D1, the primary body of the Didymos binary system, on one of the modes of the IP of

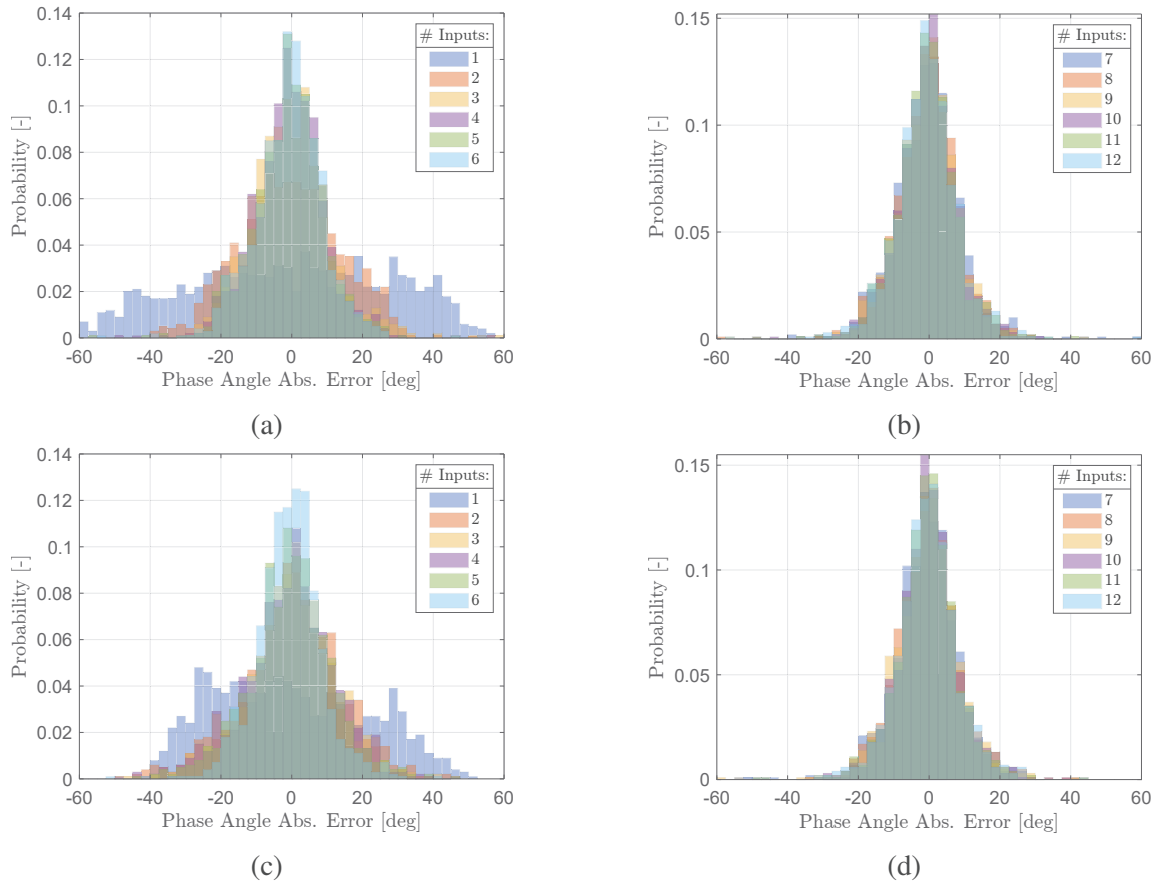


Figure 5: Histogram of the phase angle errors for a different number of inputs in the aPC. In (a) and (b) PCA is performed on all the IP features while in (c) and (d) on a subset of IP features equal to the number of the prescribed aPC inputs.

Milani. After the visit of DART, a preliminary update of the shape model estimated a 22% change in s_z, the scale across the principal axis passing by the poles of D1. This unforeseen change exceeded the 5% value considered from previous radar observations and assumed during the design of the IP and caused D1 to be more oblate than expected. In this work, the authors have characterized how different values of s_z affect the performance of the original version of the IP of Milani and determined that the cause of the performance drop is due to the particular interaction between the ellipse fitting procedure with an oblate object, which in the original version of the IP of Milani affects the estimate of the phase angle.

Additional features have been extracted from the image and a PCA analysis has been performed to identify the most relevant ones that can be used to establish a relationship with the phase angle. Of the different methods considered, the CNN performed the best, hinting that added filtering capabilities played a role in generating more successful feature vectors. On the other hand, PCE has demonstrated to be an excellent alternative even outside its traditional field of application, retaining similar performance with respect to the previous implementation of the IP and against NN approaches, that however are not fully explainable. It is also noted that the PCE implementation was limited to 3rd-order polynomials with symmetrical expansions. Differential expansions could have provided better performance and are left for future studies. Ultimately, PCE will be considered for onboard implementation given its simplicity, flexibility, and satisfactory level of performance compared with the other methods analyzed.

Finally, it is emphasized that the full datasets with $s_z = 0.78 s_0$ and $s_z = s_0$ are made publicly

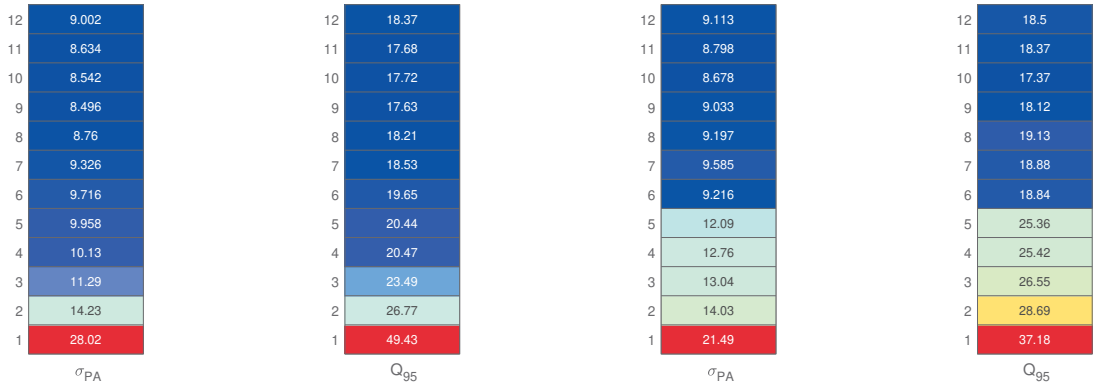


Figure 6: Heatmaps for standard deviation and 95-ile of phase angle error for a different number of inputs in the aPC. In (a) and (b) PCA is performed on all the IP features while in (c) and (d) on a subset of IP features equal to the number of the prescribed aPC inputs.

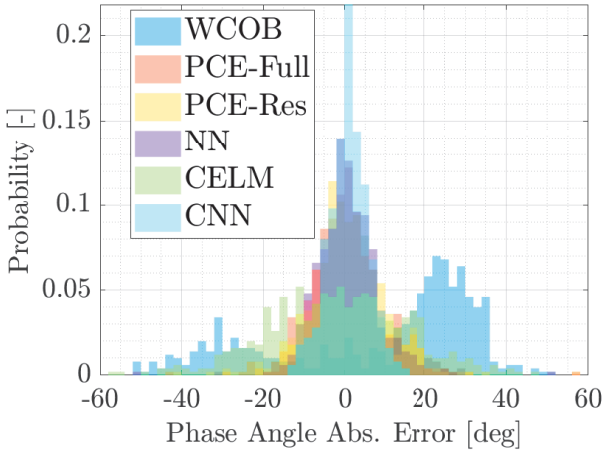


Figure 7: Histogram comparison between the methods considered in this work for the estimation of Ψ.

available for the interested readers at the following link together with the values illustrated in the results section, and expansion coefficients for the aPC and PCE: https://zenodo.org/record/7962714.

6 ACKNOWLEDGMENT

Mattia Pugliatti and Francesco Topputo 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.

APPENDIX

In this section, the coefficients for the PCE-based methods are listed. In Table 4 is possible to see the polynomial coefficients for the aPC basis from the 1st to the 9th principal components while the PCE coefficients are illustrated shortly after. All coefficients are also available in *mat* format in the *PCEFull9.mat* variable that can be downloaded from https://zenodo.org/record/7962714.

Table 4: Polynomial coefficients for the aPC basis from the 1st to the 9th principal components with $k_0 t_0 = 1$ while all other values not listed are 0.

N	1	2	3	4	5	6	7	8	9
$\mathbf{k_1 t_0}$	-13.5569	-2.70314	-5.71644	-11.2045	-3.37058	-6.30749	-24.4469	-11.8573	-15.5978
$\mathbf{k_1 t_1}$	1.66103	-3.61919	5.37342	10.708	-12.8744	-15.4522	20.1465	22.0563	-27.6226
$\mathbf{k_2 t_0}$	171.724	7.08328	16.8619	37.1778	4.22815	16.108	86.6293	47.9511	86.2179
$\mathbf{k_2 t_1}$	-41.7485	20.1739	-31.6487	-69.2988	27.354	79.8323	-156.745	-177.172	314.618
$\mathbf{k_2 t_2}$	2.52353	12.59	14.4096	32.017	39.3329	96.4759	70.2225	162.492	285.596
$\mathbf{k_3 t_0}$	-1972.28	-15.0084	-45.3044	-155.854	-5.1307	-33.9968	-240.88	-190.194	-338.869
$\mathbf{k_3 t_1}$	719.989	-67.9534	128.468	396.115	-53.5162	-240.801	862.591	1015.75	-2030.82
$\mathbf{k_3 t_2}$	-87.2702	-92.4898	-115.802	-322.954	-154.628	-536.786	-936.301	-1767.48	-3973.17
$\mathbf{k_3 t_3}$	3.51218	-38.4406	33.1872	83.3182	-111.775	-374.261	320.37	999.161	-2547.25

These are the PCE coefficients in lexographical ascending order (from 0 to 219):

0.0261, -0.011, -0.012, 0.00595, 0.0524, -0.0117, 0.0113, -0.0132, -0.00833, -0.00503, 0.17, 0.00155, 0.0106, 0.0357, -0.00359, 0.0164, -0.147, -0.00344, -0.00737, 0.0252, 0.00131, 0.0256, -0.0171, 0.00486, 0.00731, 0.0109, -0.0276, -0.00524, -0.0144, -0.0202, -0.0396, 0.0139, -0.0027, 0.00239, -0.0051, 0.0259, 0.0221, -0.00297, -0.000336, 0.00394, 0.00363, -0.00455, -0.00179, -0.00232, -0.0126, -0.0529, -0.00191, 0.00609, -0.000579, 0.0106, -0.0188, -0.00188, -0.000987, -0.0267, 0.00124, -0.0545, -0.184, -0.0107, 0.016, -0.00779, 0.0036, 0.00912, -0.00804, -0.007, -0.000509, -0.00609, -0.0168, -0.00953, -0.00318, -0.0145, 0.00915, -0.000737, -0.00313, 0.000443, 0.00271, 0.000266, 0.0148, -0.193, -0.0415, -0.00426, -0.0195, -0.0117, 0.00652, -0.119, -0.0402, 0.023, -0.00337, -0.00498, 0.00774, 0.00537, -0.013, -0.0136, -0.0122, 0.0187, -0.0177, 0.000263, -0.00396, 0.00799, 0.0143, -0.0105, 0.000795, -0.00481, 0.0118, 0.0114, 0.0106, 0.0271, -0.00904, -0.00273, 0.00855, 0.00695, 0.00418, 0.00465, -0.00257, -0.00961, -0.0176, 0.0264, 0.0197, 0.0178, 0.033, 0.0148, -0.359, 0.025, 0.0128, 0.0248, 0.0077, -0.0125, -0.00818, -0.0048, 0.00884, 0.00333, -0.0163, -0.00975, -0.0114, 0.00202, 0.00914, -0.00655, 0.00161, -0.00564, 0.00471, 0.00591, 0.013, -0.0478, -0.0142, 0.00135, -0.00371, 0.00679, 0.000787, -0.0389, -0.0837, -0.014, 0.00746, -0.0237, -0.0153, -0.0254, -0.0132, 0.00503, 0.0135, -0.0372, -0.0226, -0.0236, 0.0332, 0.0174, 0.0594, 0.0232, 0.0785, -0.132, 0.0116, 0.0159, 0.0122, 0.00156, 0.0122, -0.0255, -0.00238, -0.00191, 0.015, -0.0161, -0.00358, -0.00114, -0.000936, 0.000418, -0.00953, 0.00129, 0.0053, -0.00124, 0.00357, -0.00638, -0.0123, -0.00996, 0.00513, -0.00416, 0.00802, 0.002, -0.0349, -0.0132, -0.00646, 0.00548, -0.000944, 0.00447, 0.00201, -0.003, 0.0217, 0.00439, -0.0286, -0.0086, -0.0116, 0.021, 0.01, 0.0249, -8.68*1e-05, 0.0746, 0.026, -0.0058, -0.00796, -0.0124, 0.000401, -0.00069, 0.0143, -0.00145, 0.0535, -0.016

REFERENCES

[1] D. Xiu and G. E. Karniadakis, "The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations," *SIAM Journal on Scientific Computing*, vol. 24, no. 2, pp. 619–644, 2002. DOI: 10.1137/S1064827501387826.

[2] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Machine learning*, vol. 53, pp. 23–69, 2003.

[3] D. Xiu, *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press, 2010.

[4] 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%5C_icmlpaper.pdf.

[5] S. Oladyshkin and W. Nowak, "Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion," *Reliability Engineering & System Safety*, vol. 106, pp. 179–190, 2012. DOI: 10.1016/j.ress.2012.05.002.

[6] B. A. Jones, A. Doostan, and G. H. Born, "Nonlinear propagation of orbit uncertainty using Non-Intrusive Polynomial Chaos," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 430–444, 2013. DOI: 10.2514/1.57599.

[7] 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, doi:10.1109/mci.2015.2405316.

[8] Y.-z. Luo and Z. Yang, "A review of uncertainty propagation in orbital mechanics," *Progress in Aerospace Sciences*, vol. 89, pp. 23–39, 2017. DOI: 10.1016/j.paerosci.2016.12.002.

[9] ESA, *Hera didymos reference model, issue 5, revision 4*, Issue 5, Revision 4, 2020.

[10] S. Naidu, L. Benner, M. Brozovic, *et al.*, "Radar observations and a physical model of binary near-earth asteroid 65803 didymos, target of the dart mission," *Icarus*, vol. 348, p. 113 777, 2020, ISSN: 0019-1035. DOI: https://doi.org/10.1016/j.icarus.2020.113777.

[11] A. S. Rivkin, N. L. Chabot, A. M. Stickle, *et al.*, "The double asteroid redirection test (dart): Planetary defense investigations and requirements," *The Planetary Science Journal*, vol. 2, no. 5, p. 173, Aug. 2021, doi: 10.3847/PSJ/ac063e.

[12] 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, doi:10.3390/informatics8020033.

[13] JHUAPL, *Design reference asteroid*, Issue 4, Revision 1, 2022.

[14] M. Liotta, "Ballistic capture corridors design via polynomial chaos expansion," Politecnico di Milano, 2022.

[15] P. Michel, M. Küppers, A. C. Bagatin, *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, doi: 10.3847/psj/ac6f52.

[16] F. Piccolo, M. Pugliatti, P. Panicucci, and F. Topputo, "Toward verification and validation of the milani image processing pipeline in the hardware-in-the-loop testbench tinyv3rse," in *44th AAS Guidance, Navigation and Control Conference*, 2022, pp. 1–21.

[17] M. Pugliatti and F. Topputo, "Design of convolutional extreme learning machines for vision-based navigation around small bodies," *Journal of Guidance, Control, and Dynamics*, vol. Submitted, pp. 1–29, 2022.

[18] M. Pugliatti, V. Franzese, A. Rizza, *et al.*, "Design of the on-board image processing of the Milani mission," in *2022 AAS GN&C Conference*, ser. Breckenridge, Colorado, 2022, pp. 1–20.

[19] M. Pugliatti and F. Topputo, "Boulders identification on small bodies under varying illumination conditions," in *3rd Space Imaging Workshop, Georgia, Atlanta*, Oct. 2022, pp. 1–12.

[20] M. Pugliatti, F. Piccolo, A. Rizza, V. Franzese, and F. Topputo, "The vision-based guidance, navigation, and control system of hera's milani cubesat," *Acta Astronautica*, vol. 210, pp. 14–28, 2023, ISSN: 0094-5765. DOI: https://doi.org/10.1016/j.actaastro.2023.04.047. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094576523002205.