BOULDERS IDENTIFICATION ON SMALL BODIES UNDER VARYING ILLUMINATION CONDITIONS

Mattia Pugliatti[1*] and Francesco Topputo[1]; [1] Politecnico di Milano, Department of Aerospace Science and Technology, Via La Masa 34, 20156, Milan, Italy. [*][mattia.pugliatti@polimi.it]

Abstract. *The capability to detect boulders on the surface of small bodies is beneficial for vision-based applications such as navigation and hazard detection during critical operations. This task is challenging due to the wide assortment of irregular shapes, the characteristics of the boulders population, and the rapid variability in the illumination conditions. The authors address this challenge by designing a multi-step training approach to develop a data-driven image processing pipeline to robustly detect and segment boulders scattered over the surface of a small body. Due to the limited availability of labeled image-mask pairs, the developed methodology is supported by two artificial environments designed in Blender specifically for this work. These are used to generate a large amount of synthetic image-label sets, which are made publicly available to the image processing community. The methodology presented addresses the challenges of varying illumination conditions, irregular shapes, fast training time, extensive exploration of the architecture design space, and domain gap between synthetic and real images from previously flown missions. The performance of the developed image processing pipeline is tested both on synthetic and real images, exhibiting good performances, and high generalization capabilities*

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] . These bodies display great variability in terms of shapes and surface morphological characteristics, which pose new challenges for optical-based systems, especially under varying illumination conditions. Within the broader field of spacecraft autonomy, the specific capability to navigate around a known celestial body and comprehend its surroundings is of paramount importance to enable any autonomous decision-making process onboard a spacecraft. When considering the proximity environment of a small body and all the 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 performance with cost-effective hardware. The inclusion of such an understanding of the surrounding environment enables autonomous systems to operate at a fraction of the cost that would be traditionally required to operate with human-in-the-loop approaches, also unlocking the capability to perform enhanced critical operations unaffected by time-delayed communications[1] .

Within this context, robust identification of boulders on the surface of celestial bodies has implications both for features-based navigation techniques[2–4] and for hazard detection and avoidance for landing applications.[5–9] In this work, this is approached as a semantic segmentation task.

Previous works exist in the literature that uses image segmentation in space-related applications. In[10–13] it is used as a means to classify and distinguish geological properties of the terrain while in[14] to enhance scientific return during flybys. With the progress of artificial intelligence, and in particular deep-learning, architectures for image-segmentation applications have boomed. Most notably, outside the space domain, in[15] a new successful architecture referred to U-shaped Network (UNet) is introduced for biomedical segmentation. This architecture has later been extensively used for its design and implementation simplicity in works such as in[5–9] to generate hazard maps for safe landing site selections on the surface of celestial bodies such as the Moon, Mars, and small bodies. All the aforementioned works based on UNet architectures suffer the disadvantages typical of deep-learning architectures: computationally expensive networks which require a large amount of time for training and a large amount of labeled data. The latter poses a critical issue for the development and benchmark within the IP community of any data-driven algorithm.[2] Previous work by the authors of this paper has also been performed about semantic segmentation on the surface of small bodies. In particular in,[16] where a set of UNet architectures are designed to map the image content of small bodies into 5 different classes: background, surface, craters, boulders, and terminator region. These segmentation maps have also demonstrated to be usable for optical navigation in.[4]

An opposite trend exists in parallel that pushes for the use of shallower networks, which exploit randomization together with alternative training strategies in order to expedite the training and reduce the burden of computational complexity. Within these premises, Extreme Learning Machine (ELM)[17–19] and its subsequent evolution to hierarchical pooling architectures referred to Convolutional Extreme Learning Machine (CELM)[20–23] , have proposed a possible solution with limited degradation in performance compared to deep counterparts such as neural networks and convolutional neural networks.

The work presented in this paper is a spin-off between the one in[16] and the one in[23] with the three main following innovations. First, contrary to[16] which focuses on five layers, in this work the focus is solely put on the boulders and their robust segmentation in face of varying illumination conditions. Second, concepts from CELM theory[20,21,23] that have been successfully tested for other

IP tasks, are extended to use for image segmentation to speed up the design of the final IP pipeline. Third, particular care is set to the design of large synthetic datasets of image-label pairs through realistic artificial environments. In a push to encourage the IP community to benchmark different approaches together, these datasets are made publicly available[24] while their statistics are discussed in detail in[25] . Exploiting these three pillars, the work presented in this paper outline the design of an efficient IP pipeline for robust boulder segmentation on the surface of small bodies under varying illumination conditions.

The rest of the paper is organized as follows. First, the methodology is discussed, with particular care on the dataset's key characteristics and the design of the IP pipeline used for segmentation. Then, the results of different sections of the pipeline are presented. Finally, some conclusions and future works are discussed.

Methodology. The methodology is divided into two subsections, the first addressing the high-level characteristics of the datasets used in this work and the second illustrating the multi-step training strategy that produces the IP pipeline to segment boulders.

Datasets. The data-driven IP pipeline designed in this work necessarily requires extensively annotated datasets about boulders on the surface of small bodies seen from varying illumination conditions. Since the remarkable lack of publicly available datasets[2] from previously flown missions, artificial environments are specifically designed in this work to generate large amounts of synthetic labeled images. To do so, Blender[1] is used due to its simplicity, extensive prior usage, large support community, and open-source licensing. Using these artificial environments, three main datasets have been generated specifically for this work, for simplicity referred to as DS_1, DS_2, and DS_3. A detailed description of the setup used to generate them as well as a statistics characterization is presented in detail in[25] . This section focuses on a brief overview of the main characteristics of each dataset.

DS_1 is composed of synthetic images of single instances of boulders positioned on a procedural-varying quasi-spherical surface. Its main purpose is to represent a single instance of a boulder positioned on the surface of a generic small body. DS_2 is also composed of synthetic images, however, multiple instances of boulders are scattered across the surface of an enhanced shape model of the primary body of the (65803) Didymos asteroid. This is done to represent realistic boulder distributions scattered across a generic regular shape body. Finally, DS_3 is composed of a small set of real images manually labeled from previously flown missions toward asteroids (25143) Itokawa,[26] (162173) Ryugu,[27] and (101955) Bennu.[28] Its purpose is to represent real boulder populations scattered across the surface of existing small bodies.

The image-label pairs of DS_1 are created using a unitary radius high-resolution spherical mesh to represent the small body, while boulder meshes are generated randomly using the *Rock generator* add-on in Blender. A set of 30 meshes is used to represent archetype shapes of boulders, which are then singularly positioned on the surface of the body with random orientation, scaling, and albedo. In order to simulate camera positions, a random cloud of points is generated around the boulder, while illumination conditions are also varied randomly. During acquisition the attitude is assumed to be ideal, pointing towards the center of the boulder. Images are rendered at a resolution of 256×256, but are then post-processed with random cropping to 128×128 size images and the addition of artificial noise. Post-processing is fundamental in making sure the boulders are not always centered in the images. Both boulders and surfaces are simulated utilizing an Akimov scattering law implemented in the shading tab of Blender. At each acquisition, the characteristics of the surface of the body are varied randomly to simulate different roughness that disturbs the environment surrounding each boulder. With this setup, a total of 45269 image-label pairs are rendered for DS_1. Note that the masks of the boulder and surface are obtained thanks to the *Cycles* rendering engine in Blender and are generated both with and without shadows. These are later split into training, validation, and test sets as illustrated in Table 1. Figure 1 represents a sample of image-label pairs of DS_1 after rendering and after post-processing.

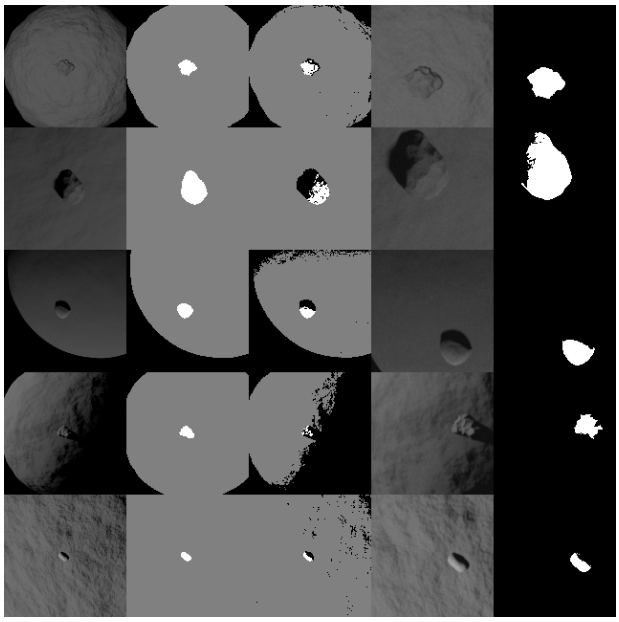

Figure 1. Sample of image-label pairs of DS_1***. From left to right:*** 256×256 ***grayscale rendering in Blender, masks without shadows, masks with shadows, followed by*** 128×128 ***noisy and randomly cropped grayscale images, and relative masks with shadows.***

The procedure adopted to generate the image-label

[1]https://www.blender.org/, retrieved 13th of September, 2022.

pairs of DS_2 is in part similar to the one illustrated for DS_1. The main differences are in the number of boulders positioned on the surface of the body, the size of the rendered images in Blender (128×128), and the lack of random cropping during post-processing (only artificial noise is added to the rendered images). During rendering, instead of placing a single boulder, multiple ones are positioned on the surface of the enhanced Didymos shape model to represent a realistic boulders distribution. Once again, as in DS_1, both lighting, scale, albedo, and intensity variations are randomly implemented to obtain a generalized dataset. DS_2 is made of 35183 image-label pairs. Their split into training, validation, and test sets is summarized in Table 1, while a sample of image-label pairs is visible in Figure 2.

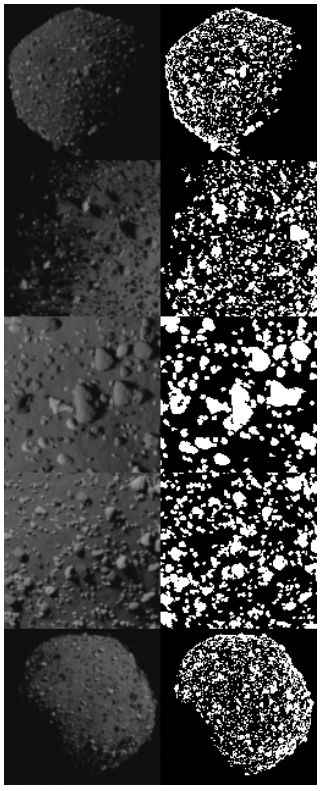

Figure 2. Sample of image-label pairs of DS_2**.** 128×128 ***noisy grayscale images (left) and relative boulder masks (right).***

Finally, the DS_3 dataset is generated starting from 75, 256×256 cropped images which show clear boulders presence that has been manually labeled in[16] . Each image-mask pair is then subdivided into 4 128×128 smaller ones to reach a total of 300 samples. By design, this dataset only contains the masks of the largest boulders, as is visible in the sample in Figure 3.

Table 1 summarizes the main characteristics of the split used in the train, validation, and test sets used in this work. Note that, as explained in[25] , the main difference between Te_1 and Te_2 is that the first represents images with a balanced distribution of phase angles while the latter does not.

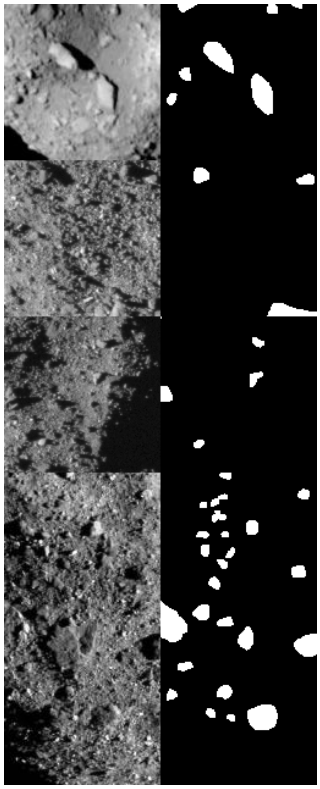

Figure 3. Sample of image-label pairs of DS_3**.** 128×128 ***real grayscale images (left) and relative boulder masks (right), manually labeled in.[16]***

Name	Acronym	DS_1	DS_2	DS_3
Training	Tr	30181	20095	-
Validation	V	5044	5044	-
	Te_1	5044	5044	-
Test	Te_2	5000	5000	-
	Te_3	-	-	300
Total	-	45269	35183	300

Table 1. Summary of the datasets used in this work.

Network architectures. In this work, a data-driven IP pipeline is designed to perform robust boulder segmentation under varying illumination conditions. The pipeline is designed specifically for this work following an incremental training strategy that involves different architectures. The strategy is intended to efficiently accompany the design by incrementally training portions of the final architecture that performs boulders segmentation on the surface of small bodies. The training is designed as a 4 steps process, as schematized in Figure 10, using a Tesla P100-PCIE 16Gb GPU, with a 27.3 Gb of RAM in Google colab[2].

[2]https://colab.research.google.com/, retrieved 13th of September, 2022.

In the first two steps, images from DS_1 are used together with the CoB of the boulders masks with shadows to design an encoder structured as a Hierarchical Pooling Network (HPN). First, to select the proper architecture of the HPN, the CELM framework[20–22] is selected as an effective tool to efficiently explore the architecture design space. In CELM theory training of a HPN happens by solving a regularized least-square problem that finds the best set of weights $\boldsymbol{\beta}$ connecting the last hidden layer of the architecture with its output layer. All remaining weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ of the network are set randomly at initialization and are kept frozen during training. To find out $\boldsymbol{\beta}$ means to solve the following problem[20] :

$$\text{Minimize} : \|\boldsymbol{\beta}\|_2^2 + C\,\|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|_2^2 \tag{1}$$

where $\boldsymbol{H}_{N\times L}$ is the hidden layer output matrix generated by an input tensor of depth N that passes through the network up to the last dense layer before the output layer (which in this work is the fully connected layer made by L neurons), $\boldsymbol{T}_{N\times M}$ is the target output layer, where M is the number of labels to predict for each image. The addition of the regularization term C and the minimization of the norm of $\boldsymbol{\beta}$, is proven to increase stability and generalization of the network.[18] The solution of Equation 1 is:

$$\boldsymbol{\beta} = \begin{cases} \boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{T}, & \text{if } N \le L \\ \left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}^T\boldsymbol{H}\right)^{-1}\boldsymbol{H}^T\boldsymbol{T}, & \text{if } N > L \end{cases} \tag{2}$$

depending on whether it is more convenient to invert an $N \times N$ or $L \times L$ matrix. In this work, the validation set is used during training to select the best regularization parameter C, while the training set is used to determine the best set of $\boldsymbol{\beta}$. Since training happens by solving a non-iterative regularized least square problem and since a single passage of the input tensor is required to compute $\boldsymbol{H}$, training time is extremely short. Such property enables a fast and efficient exploration of the architecture design space, as illustrated in[21,23] .

In this work, the CELM training capabilities are exploited to select the best-performing architecture for an encoder that will be part of the final architecture. By predefining, a set of rules to build up a HPN as an encoder for the prediction of the boulders CoB, the architecture with the best capacity is found. As schematized in Figure 4 the encoder is designed by a sequence of cells C_i which operate on batches of tensors to generate other tensors. Each cell is composed of a combination of dilated convolutions, activation functions, and pooling layers, as exemplified in the top part of Figure 4. In particular, dilated convolutions are inserted into the encoder for their beneficial effects[12,29] in augmenting the receptive field of the kernels as well as their capability to boost segmentation performance. In this work, dilated convolution with rates 1, 2, and 3 are used and then stacked together to produce the output tensor.

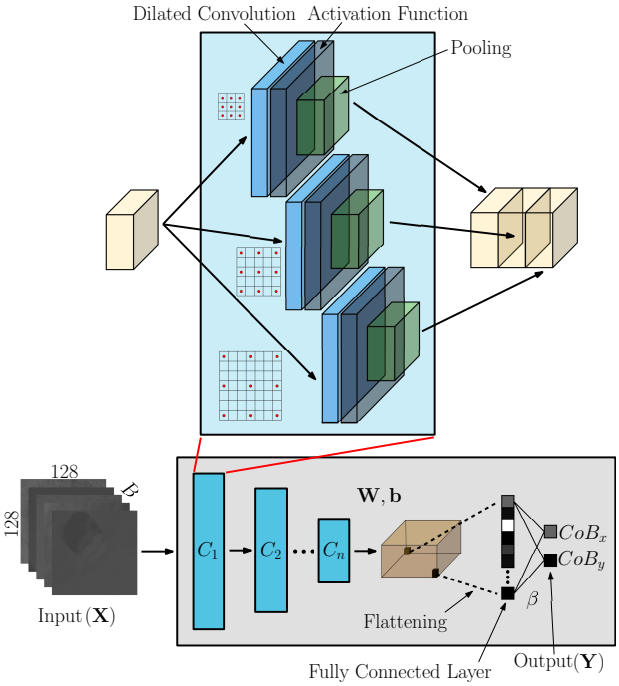

Figure 4. Schematic of the encoder used in this work.

By assuming a constant kernel size of 3×3 and an exponential depth expansion coefficient equal to 2, different architectures are generated by varying the pooling strategy P, the initial depth of the network d_0, the total number of cells n, the activation functions A, the kernel initialization strategy K_d, and the number of random runs for each architecture N_r. The regularization parameter C is varied for each architecture as $10^{-3}, 10^{-2}, 10^{-1}, 1, 10^1, 10^2, 10^3$. By combining together these parameters a total of 1134 different architectures are generated. The setup that achieves the best performance in predicting the boulder's CoB is represented in bold in Table 2, which is achieved using $C = 1$.

Name	Values
P	mean, **max**
d_0	4, 5, **16**
n	$(3,4,5)_{d_0=4}$, $(4,\mathbf{5},6)_{d_0=8}$, $(5,6,7)_{d_0=16}$
A	NReLU, ReLU, LReLU, **ELU**, tanh, sigmoid, none
K_d	RandomUniform (-1,1), RandomNormal (0,1), **Orthogonal**
N_r	3

Table 2. Summary of the hyper-parameters used in this work to search for the optimal architecture of the encoder.

In Figure 5 it is possible to visualize how the combination between the hyper-parameters in Table 2 influences the performance of the HPN in its 1134 combinations.

In this work, the best-performing architecture out of

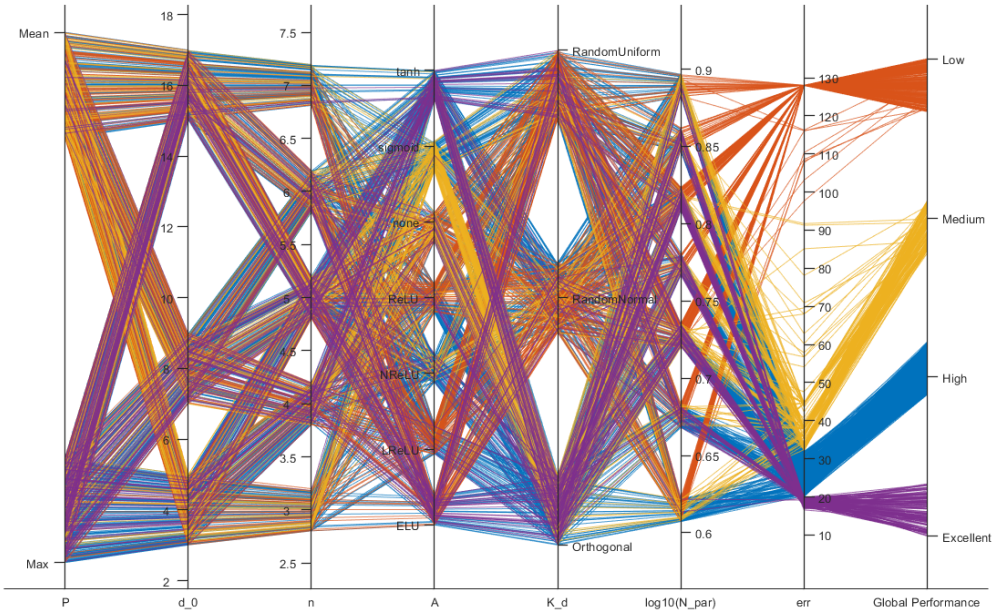

Figure 5. Parallel-plot showing the dependencies between the different hyper-parameters illustrated in Table 2 and the network performances. The lines are colored by 4 different quality metrics related to the performance (excellent, high, medium, and low).

the pool of the one tested has been chosen for implementation and is generally referred to as the CELM-encoder. However, the approach presented in this paper, coupled with representative global metrics of interest, can be used to select families of architectures that are expected to work well for a given task. The total time needed to train the 1134 architectures with the CELM paradigm is equivalent to 48.3 hours. On average, 13.93% of the time is spent on the forward pass of the validation tensor, 81.32% on the forward pass of the training tensor, while the remaining 4.75% is spent solving equation 2. The time saved exploring the architecture design space with CELMs is crucial to fast forward and ease the training in the following steps.

In the second step of the training, the single best performing HPN trained with the CELM paradigm is re-trained using mini-batch gradient descent[30] such as in the case of a traditional CNN. During this training, the architectural elements are frozen while the weights and biases of the kernels are optimized starting from various batch sizes (64, 128, 256, 512) and learning rates(10^{-4}, 10^{-3}, 10^{-2}). Each setup is initialized and run randomly 2 times for a total of 24 training instances for short epochs. The best-performing ones are selected based on the best error achieved over the validation split during the entire training and are re-run while increasing the epochs. The loss function used to train the network is the mean squared error. The final setup is achieved using a batch size B of 32 samples, with a learning rate lr of 10^{-4}, and a dropout rate in the fully connected layer of 0.2.

Note that the training time of the best performing architecture for 200 epochs using the mini-batch gradient descent method required a total of 9504.7s, while the equivalent training time with a CELM would take on average 153s per architecture (spent mostly in the forward pass to generate $\boldsymbol{H}$ for the training and validation sets). The total training time to find out the best combination of learning rate and batch size in $step_2$ is roughly 24h. Should all the 1134 architectures have been trained using the mini-batch gradient descent, considering the encoder as a CNN, the exploration of the architecture design space would have resulted in a much more computationally expensive process. The combination between CELM and CNN training paradigms allows thus for efficient exploration of the architecture design space and thus of the encoder's design. The detailed architecture of the CELM and CNN encoders is represented in Figure 6 using TensorFlow 2.10 notation.

The third step of the training exploits the CNN-encoder refined from the previous step and inserts it into a larger architecture configured for segmentation. For such a task, a UNet[15] is considered, as it has shown good performance and generalization capabilities[16] with small body images. A schematic of the UNet architecture is represented in Figure 7. Its main characteristics are: an encoder-decoder setup that resembles a U-shape, the lack of a fully connected layer, the presence of skip connections after the activation layer of each cell which are copied and concatenated to their corresponding layers in the decoder, and the fact that training only involves the decoder while the encoder is kept frozen as it already possess encoding capabilities acquired from previous training on other tasks.

Similarly to,[16] the UNet is trained by incrementally

Figure 6. Detailed architecture of the encoder, made of 3'527'040 parameters.

increasing the epochs while testing various combinations of dropout values, batch size, learning rate, and depth of the decoder layers. A Weighted Sparse Categorical Cross Entropy (WSCCE)[16] is used as a loss function, while the Mean Intersection Over Union (MIOU) is used as a metric. The weights for the WSCCE loss are computed from statistical analysis of the pixel content in the masks of the training set of DS_1. As 3.99% of the pixels are boulders while 93.01% are not, the complement of these values are used respectively as weights of the non-boulder

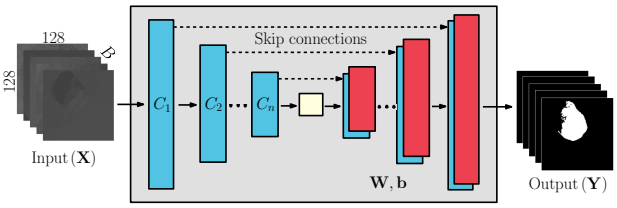

Figure 7. Schematic of the UNet architecture used in this work.

and boulder classes. The best performing architecture has been found with a dropout equal to 0.2, a batch of 256, a learning rate of 0.001, and decoder depths of 192, 96, 48, and 24. The total training time spent to find out this setup is equal to 18.9 hours. The detailed UNet architecture is illustrated in Figure 8 using TensorFlow 2.10 notation.

Finally, in the fourth and last step of the training the same procedure illustrated in the previous step is repeated but using the DS_2 dataset. In this case, the training of the UNet is not initialized from scratch, but starts from the set of weights and biases found in the previous step, to help the network to plateau at higher values of the MIOU on the validation set, thus achieving better performance and generalization capabilities that it would from starting from scratch. The WSCCE loss function uses weights corresponding to 27.52% and 72.48% respectively for the non-boulder and boulder classes. The final architecture is trained over 400 epochs with a learning rate of 0.001 and a batch size of 16. The total training time spent to find out this setup is equal to 12 hours.

Figure 9 illustrates the entire training history of the final UNets trained in steps 3 and 4. The green points represent the events in which the maximum value of MIOU has been achieved over the validation set. When the architectures are initialized for testing, they are loaded with the sets of weights and biases that have been achieved at these epochs. It is noted that the training of the UNet with single boulders required roughly 5 hours while the one with multiple boulders 3 hours.

A schematic that summarizes the entire 4-step training procedure is illustrated in Figure 10. CELM theory is used in $step_1$ to expedite the architecture design search of an encoder, which is further refined in $step_2$ as a CNN. The encoder is trained over a regression task on the DS_1 dataset, to predict the CoB of a single boulder appearing in the image. A partial training of the UNet for segmentation is then performed in $step_3$ using DS_1, which is further refined in $step_4$ with the use of the DS_2 dataset. This incremental approach ultimately allows better generalization and improved performance of the final UNet when compared to a training executed from scratch considering only $step_4$[16] . The entire training, from top to bottom, to obtain the final IP network for segmentation took roughly 103 hours.

Results. In this section, the results of the IP networks are illustrated and discussed in detail. First, the performances of the encoders are commented, followed up by those of the UNet architectures on the single and multiple boulder cases.

Encoder networks. To train the encoder and to define its generalization capabilities on the validation and test sets, an error metric is defined as:

$$\varepsilon_{CoB} = \sqrt{(CoB_e^u - CoB_t^u)^2 + (CoB_e^v - CoB_t^v)^2} \quad (3)$$

where CoB_e^u and CoB_e^v are the estimated coordinates of the CoB respectively in the u and v axes of the image plane and CoB_t^u and CoB_t^v are the corresponding true coordinates.

The performances of the CELM and CNN encoders designed in this work are illustrated in Table 3 for the two test sets of DS_1.

Encoder	Dataset	$\mu(\varepsilon_{CoB})$ $[px]$	$\sigma(\varepsilon_{CoB})$ $[px]$
CELM	Te1	16.36	11.46
CELM	Te2	16.30	11.36
CNN	Te1	7.01	7.30
CNN	Te2	7.06	7.40

Table 3. Encoders performances on the test sets of DS_1.

It is possible to see that the error is in the same order of magnitude between the two encoders, the CNN being the best performing one. It is clear that the mini-batch gradient descent optimization involving all weights and biases of the kernels in the convolutional layers favors a better-performing network. Moreover, it is also observed that both encoders exhibit limited variability between Te_1 and Te_2.

In Figure 11 and Figure 12, the histograms of the distributions of $\boldsymbol{\beta}_u$, $\boldsymbol{\beta}_v$, $\boldsymbol{W}_u$ and $\boldsymbol{W}_v$ between the last hidden layer and the output layer of the CELM and CNN encoders are illustrated. Both sets of weights are normally distributed, however it is noted that those of the CNN exhibit a variance that is one order of magnitude smaller. Finally, it is also observed that while the CELM encoder does not possess a bias term in the connection between the two layers, the CNN does and its value is very similar between its two components (0.148).

Segmentation networks. The performances of the best UNet architecture generated after $step_3$ of the overall training procedure are summarized in Table 4 on the test sets of DS_1 in terms of WSCCE, Accuracy (A), and MIOU. As for the encoder case, the network performs similarly in Te_1 and Te_2, exhibiting very high values of MIOU. It is noted that the values achieved by such a network are an improvement of the ones in,[16] where the segmentation network is tasked to predict a 5-layer mask instead of the 2-layer one of this work. It is also observed that the network is capable to predict a single boulder's

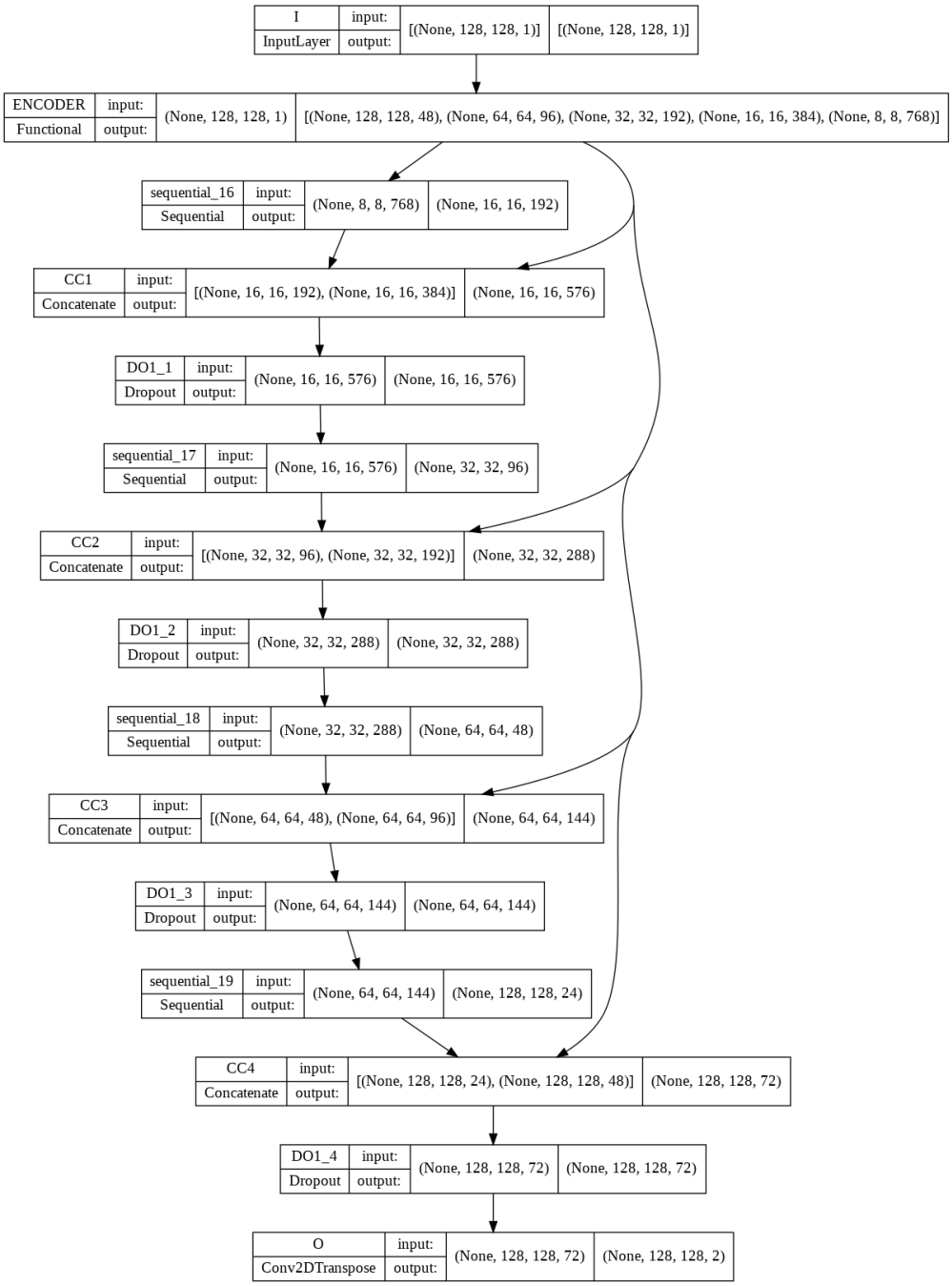

Figure 8. Detailed architecture of the UNet, made of 5'510'066 parameters, 1'982'306 of which are trainable.

presence robustly under a variety of illumination conditions.

Dataset	$\mu(WSCCE)$ [−]	$\mu(A)$ [%]	$\mu(MIOU)$ [%]
Te1	7.4 10^{-3}	99.19	90.78
Te2	9.6 10^{-3}	99.20	91.03

Table 4. UNet performance on the test sets of DS_1.

Similarly, Table 5 summarizes the performances of the UNet trained in $step_4$ with images of multiple boulders on the test sets of DS_2 and DS_3. Indeed, the presence of a large population of multiple boulders seems to challenge the performance of the UNet, which is lower than the one trained with single boulders in $step_3$. This is also illustrated by the WSCCE (which is two orders of magnitude higher than in the previous case) as well as from

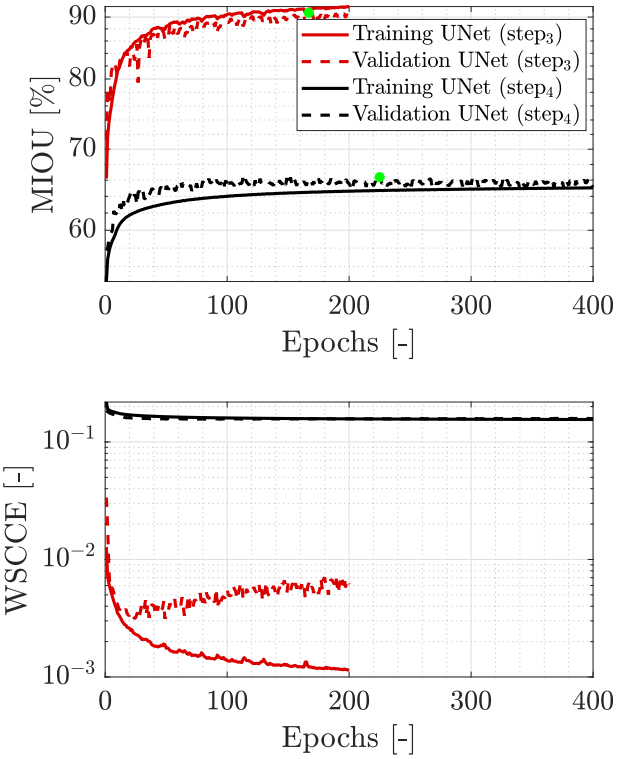

Figure 9. Training history of the MIOU (top) and WSCCE (bottom) of the UNet trained in step$_3$ and step$_4$.

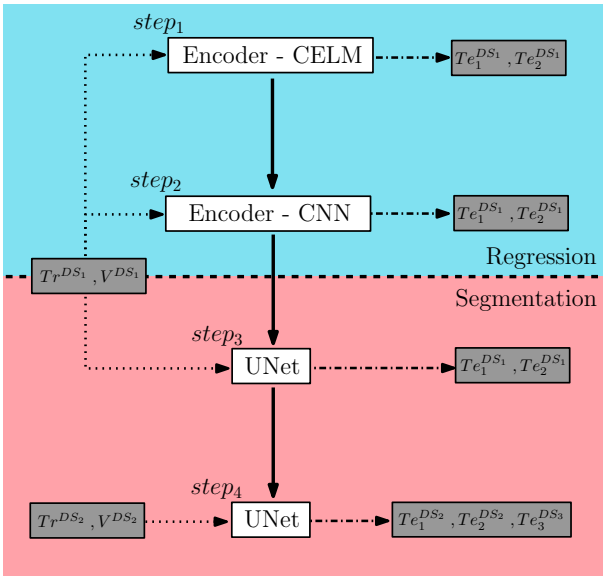

Figure 10. Schematic of the whole training procedure used in this work.

the values of A and $MIOU$.

It is also commented that the current masks in DS_3 demonstrated to be inappropriate for the specific design in this work. These masks were originally designed in[16] by manual labeling of the most prominent boulders in 256 × 256 images. However, the UNet developed in this work is capable of predicting much more boulders than the ones represented by the ground truth masks in DS_3, making them unreliable for quantitative analyses. Nonetheless, images from this dataset turned out to be useful for a qualitative assessment.

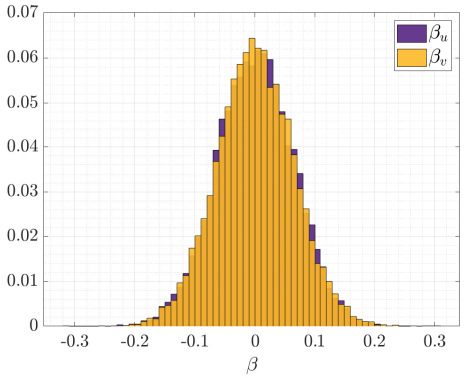

Figure 11. Distribution of the weights β in the last layer of the CELM-encoder. The y-axis shows the relative probability of each bin.

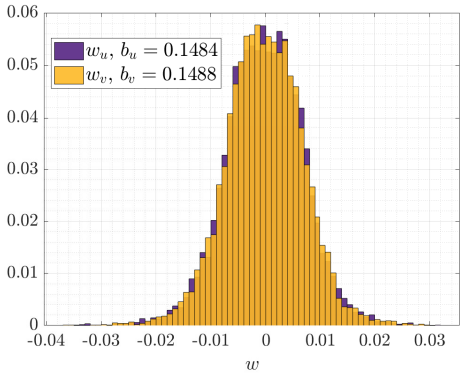

Figure 12. Distribution of the weights w in the last layer of the CNN-encoder. The y-axis shows the relative probability of each bin.

Figure 13, Figure 14, and Figure 15 showcase random samples of input images, true and predicted masks by the UNet on test images of Te_1 and Te_3 of DS_1, DS_2, and DS_3.

The very high performances of the network summarized in Table 4 are reflected in the well-predicted masks in Figure 13. Such capabilities are also successfully transferred to the UNet developed afterward with multiple instances of boulders. As it is possible to see in Figure 14, the network is capable to predict correctly a large amount of boulders on the surface of Didymos with varying geometric and illumination conditions. It is also noted that the true masks exhibit challenging conditions in which a dense boulder's presence makes the surface almost entirely covered. This somehow reflects real environmental conditions, such as in Ryugu and Bennu. From the pre-

Dataset	$\mu(WSCCE)$ [–]	$\mu(A)$ [%]	$\mu(MIOU)$ [%]
Te1	1.59 10^{-1}	82.22	66.09
Te2	1.6 10^{-1}	81.98	66.04
Te3	2.8 10^{-1}	62.53	33.26

Table 5. UNet performance on the test sets of DS_2 and DS_3.

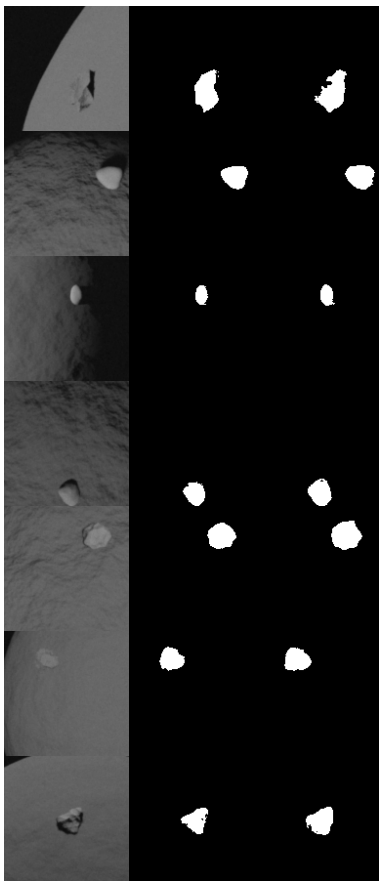

Figure 13. Samples of input image (left), true mask (center), and predicted mask (right) from the UNet trained in step$_3$ on test images of DS_1.

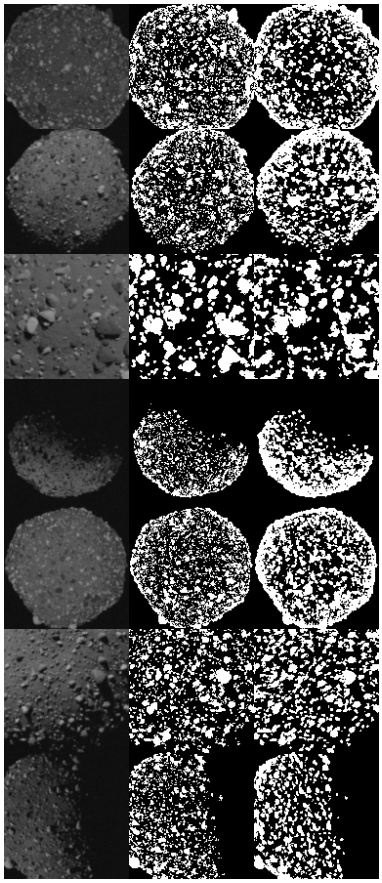

Figure 14. Samples of input image (left), true mask (center), and predicted mask (right) from the UNet trained in step$_4$ on test images of DS_2.

dicted masks in Figure 14, it is also possible to note an incorrect behavior of the network in those cases in which the body is not saturating the camera's FOV. In these cases, the network overestimates the presence of boulders over the edge, which ultimately drives down its performance. This seems to happen as the ray-tracing algorithm correctly labels true boulders over the projection of the edge in the image plane even when only a few pixels are observed over the very edge of the body, as it is also possible to observe from the true masks in Figure 14. Such a labeling mishap may have encouraged the network to actively label the entire edge as a boulder, which is a capability that is also transferred to the prediction on real images. Finally, it is also observed that the same phenomenon does not occur in the terminator region of the body, in which the boulder's labels are nullified by shadows.

Lastly, the final UNet is put to the test to predict boulders on real images from previously flown missions. It is remarked that these images have never been seen by the network during training and that they have only been seen in inference without further adjustments or fine-tuning on the network itself. First of all, as it is possible to see in Figure 15, it is commented that the noise levels from real cameras do not seem to pose a particular challenge to the network in terms of generalization. Albeit the network has been trained without a tailored noise setup to model any particular camera, the network performs similarly in all types of images considered. Is it thought that thanks to the injection of artificial noise as well as thanks to the particular care that has been put into the design of the DS_1 and DS_2 artificial datasets in their varying illumination, albedo, and scattering properties, the network is capable to generalize well enough. This is a promising feature for a direct network application on real sensor images. The only case which generated mild artifacts on images in DS_3 is visible in the 2nd case from the bottom in Figure 15.

As commented before, as it is possible to see from the

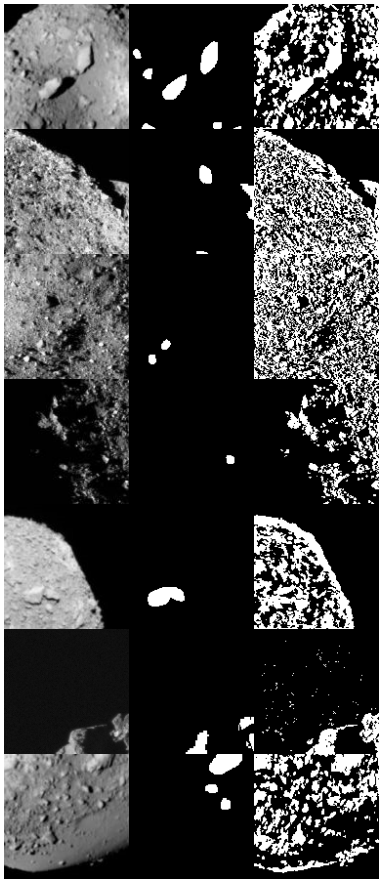

Figure 15. Samples of input image (left), true mask (center), and predicted mask (right) from the UNet trained in step$_4$ on test images of DS_3.

true masks in Figure 15, too few boulders have been manually labeled to allow a quantitative assessment of the network in this case. Nonetheless, it is noted that boulder populations seem to be detected correctly over the surface. Comparing these types of predictions with the ones in,[16] it is possible to conclude that the training presented in this work successfully pushed the architecture to detect well different-size boulders with a higher frequency than previously done. This ultimately proves to be a challenge on images of Bennu and Ryugu, since there is a risk of predicting boulder fields as uniformly spread features all over the surface. The desired behavior would be somewhat in between being able to predict small-medium size boulders on the surface as well as distinguish them clearly from large and prominent ones embedded in them. This ultimately poses a challenging problem that remains to be addressed for a real application.

Conclusions. In this work, an IP pipeline to segment boulders on the surface of small body under a variety of illumination conditions has been developed through a 4-steps incremental training strategy. This task has been possible thanks to synthetically generated datasets of image-label pairs, which are made available in[24] . The IP network developed in this work exhibited excellent performance in segmenting isolated boulders. When applied to synthetic and real images of multiple boulders, the network has also demonstrated the capability to correctly isolate boulders, with degraded performance compared to single ones. The networks also exhibited high generalization capabilities, which are deemed to be delivered by the datasets intrinsic variabilities as well as by the addition of artificial noise on the images.

Future iterations of this work will be directed toward an increase in the network generalization to a variety of small body shapes and boulder distributions. For this purpose, future updated versions of DS_2 may include different distributions for the same body as well as multiple global shape models. It has also been observed that labels of boulders on the edge of the body seem to introduce an erroneous behavior in the network. In future datasets iteration, it may be appropriate to remove such labels. The lack of labeled datasets for these types of IP tasks is ultimately a showstopper for the interested IP community for the development of data-driven algorithms. In this work, we have made all datasets but DS_3 publicly available for any interested user. It would be of interest to compare the performances of other approaches directly on the same datasets. A future collaborative effort could also be directed toward manual labeling of a sizeable chunk of real images obtained from previously flown missions. Finally, an open question remains on how to address the segmentation and identification of prominent boulders embedded into large boulder fields. This fractal nature represents a challenge in the labeling strategy presented in this work.

Acknowledgment. The authors 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. M.P. would also like to thanks Carmine Buonagura for the fruitful discussions and his help on the rendering of the datasets.

References.

[1] M. B. Quadrelli, L. J. Wood, J. E. Riedel, M. C. McHenry, M. Aung, L. A. Cangahuala, R. A. Volpe, P. M. Beauchamp, and J. A. Cutts, "Guidance, navigation, and control technology assessment for future planetary science missions," *Journal of Guidance, Control, and Dynamics*, vol. 38, pp. 1165–1186, July 2015. doi:10.2514/1.g000525.

[2] J. Song, D. Rondao, and N. Aouf, "Deep learning-based spacecraft relative navigation methods: A survey," *Acta Astronautica*, vol. 191, pp. 22–40, 2022.

[3] J. Villa, J. McMahon, B. Hockman, and I. Nesnas, "Autonomous navigation and dense shape reconstruction using stereophotogrammetry at small celestial bodies," Feb 2022.

[4] M. Pugliatti and F. Topputo, "Navigation about irregular bodies through segmentation maps," in *31st Space Flight Mechanics Meeting, Charlotte, NC*, no. AAS 21-383, pp. 1169–1189, Feb 2021.

[5] A. Scorsoglio, A. D'Ambrosio, L. Ghilardi, R. Furfaro, B. Gaudet, R. Linares, and F. Curti, "Safe lunar landing via images: A reinforcement meta-learning application to autonomous hazard avoidance and landing," in

AAS/AIAA Astrodynamics Specialist Conference 2020, vol. 175, pp. 91–110, Univelt Inc., San Diego, CA, Aug 2020.

[6] K. Iiyama, K. Tomita, T. N. Bhavi A. Jagatiaz and, and K. Ho, "Deep reinforcement learning for safe landing site selection with concurrent consideration of divert maneuvers," in *AAS/AIAA Astrodynamics Specialist Conference 2020*, vol. 175, pp. 111–130, Univelt Inc., San Diego, CA, Aug 2020.

[7] K. Tomita, K. A. Skinner, and K. Ho, "Uncertainty-aware deep learning for autonomous safe landing site selection." doi:10.13140/RG.2.2.15224.98564, 2021.

[8] 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, pp. 1–16, Feb 2022.

[9] T. Claudet, K. Tomita, and K. Ho, "Benchmark analysis of semantic segmentation algorithms for safe planetary landing site selection," *IEEE Access*, vol. 10, pp. 41766–41775, 2022.

[10] D. Thompson, S. Niekum, T. Smith, and D. Wettergreen, "Automatic detection and classification of features of geologic interest," in *Proceedings. of IEEE Aerospace Conference*, pp. 366–377, 2005. doi:10.1109/AERO.2005.1559329.

[11] 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, p. 43, Oct. 2014. 10.1088/0004-637X/794/1/43.

[12] E. Goh, J. Chen, and B. Wilson, "Mars terrain segmentation with less labels," *arXiv preprint arXiv:2202.00791*, 2022.

[13] doi:10.1016/j.cageo.2016.12.015.

[14] T. J. Fuchs, D. R. Thompson, B. D. Bue, J. Castillo-Rogez, S. A. Chien, D. Gharibian, and K. L. Wagstaff, "Enhanced flyby science with onboard computer vision: Tracking and surface feature detection at small bodies," *Earth and Space Science*, vol. 2, pp. 417–434, Oct. 2015. doi:10.1002/2014ea000042.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, vol. 9351 of *LNCS*, pp. 234–241, 2015. doi:10.1007/978-3-319-24574-4_28.

[16] M. Pugliatti and M. Maestrini, "Small-body segmentation based on morphological features with a u-shaped network architecture," *Journal of Spacecraft and Rockets*, vol. In advance, pp. 1–15, 2022. doi: 10.2514/1.A35447.

[17] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, pp. 489–501, Dec. 2006. doi:10.1016/j.neucom.2005.12.126.

[18] G.-B. Huang, "An insight into extreme learning machines: Random neurons, random features and kernels," *Cognitive Computation*, vol. 6, pp. 376–390, Apr. 2014. doi:10.1007/s12559-014-9255-2.

[19] 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. doi:10.1016/j.neunet.2014.10.001.

[20] 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, pp. 18–29, May 2015. doi:10.1109/mci.2015.2405316.

[21] 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.), pp. 1089–1096, Omnipress, 2011.

[22] 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, p. 33, May 2021. doi:10.3390/informatics8020033.

[23] 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.

[24] M. Pugliatti, "DOORS: Dataset fOr bOuldeRs Segmentation," 2022. Sept 2022, Zenodo, V1.0, doi: 10.5281/zenodo.7107409.

[25] M. Pugliatti and F. Topputo, "DOORS: Dataset fOr bOuldeRs Segmentation. statistical properties and blender setup," 2022. arXiv, September 2022, Creative Commons Attribution 4.0 International.

[26] M. Yoshikawa, J. Kawaguchi, A. Fujiwara, and A. Tsuchiyama, "Hayabusa sample return mission," *Asteroids IV*, vol. 1, pp. 397–418, 2015. doi:10.2458/azu_uapress_9780816532131-ch021.

[27] S. Watanabe, Y. Tsuda, M. Yoshikawa, S. Tanaka, T. Saiki, and S. Nakazawa, "Hayabusa2 mission overview," *Space Science Reviews*, vol. 208, no. 1, pp. 3–16, 2017. doi:10.1007/s11214-017-0377-1.

[28] D. S. Lauretta, S. S. Balram-Knutson, E. Beshore, W. V. Boynton, C. Drouet d'Aubigny, D. N. DellaGiustina, H. L. Enos, D. R. Golish, C. W. Hergenrother, E. S. Howell, C. A. Bennett, E. T. Morton, M. C. Nolan, B. Rizk, H. L. Roper, A. E. Bartels, B. J. Bos, J. P. Dworkin, D. E. Highsmith, D. A. Lorenz, L. F. Lim, R. Mink, M. C. Moreau, J. A. Nuth, D. C. Reuter, A. A. Simon, E. B. Bierhaus, B. H. Bryan, R. Ballouz, O. S. Barnouin, R. P. Binzel, W. F. Bottke, V. E. Hamilton, K. J. Walsh, S. R. Chesley, P. R. Christensen, B. E. Clark, H. C. Connolly, M. K. Crombie, M. G. Daly, J. P. Emery, T. J. McCoy, J. W. McMahon, D. J. Scheeres, S. Messenger, K. Nakamura-Messenger, K. Righter, and S. A. Sandford, "Osiris-rex: sample return from asteroid (101955) bennu," *Space Science Reviews*, vol. 212, no. 1, pp. 925–984, 2017. doi:10.1007/s11214-017-0405-1.

[29] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[30] R. Szeliski, *Computer Vision*. Springer International Publishing, 2nd ed., 2022. doi:10.1007/978-3-030-34372-9.