

ON-BOARD SMALL-BODY SEMANTIC SEGMENTATION BASED ON MORPHOLOGICAL FEATURES WITH U-NET

Mattia Pugliatti*, Michele Maestrini†, Pierluigi di Lizia‡ and Francesco Toppoto§

Small-bodies such as asteroids and comets exhibit great variability in surface morphological features. These are often unknown beforehand but can be exploited for hazard avoidance during landing, autonomous planning of scientific observations, and for navigation purposes. The detection and classification of such features is a laborious task that requires extensive manual work done by experts in the field. This step renders online usage of images unfeasible for these applications. Such limitation could be overcome thanks to the recent advances in the field of neural networks, which allow to recognize features automatically from an acquired image. However, to train such networks, an annotated dataset needs to be generated with care by field experts, thus requiring once again extensive work and human-in-the-loop. In this work, a methodology that exploits an open-source rendering software, ray-tracing masking, and simple image processing techniques is illustrated, which allows to automatize the segmentation process and build up a robust database of labeled features (i.e. background, surface, craters, boulders, and the terminator region) for small-bodies. A procedural code is designed to generate images and their labels over 7 different small-body shapes for a total of 12,550 images that are used to train a Convolutional Neural Network with a U-Net architecture in the task of semantic segmentation. The performances of the network are then analyzed in 4 different scenarios. First, the network is evaluated on a test set composed of 1,050 new images belonging to bodies seen during training. Secondly, the network is evaluated on 3,000 synthetic images from 2 models that have not been encountered in training. Afterward, one of these latter models is tested in a flyby trajectory scenario consisting of 56 images. The results of the first three tests show state of the art performances and the capability of this method to generalize features across synthetic data. Finally, the network's performances are qualitatively assessed with a set of 59 real images from previously flown missions, highlighting the current limits of this approach. These shortcomings suggest possible directions for future improvement, which are discussed in this work.

INTRODUCTION

Small-bodies such as asteroids and comets are characterized by their strong irregularities in shape, physical properties, orbital characteristics, composition, and surface morphological features. The latter are often not characterized in detail or cannot be defined at all from ground observations, which is especially the case for craters and boulders. These features, however, could provide important information to be used in autonomous on-board applications. The task to determine the type of features in an image is referred to as image semantic segmentation. Several approaches have been developed to perform or exploit image segmentation for space applications. For example, in [1] probabilistic fusion methods are developed for segmentation, detection, and classification of geological properties for a martian rover application. Instead, in [2], a series of advanced image processing techniques for enhanced flyby science are introduced. Amongst them, [2] introduces a methodology for autonomous features detection supported by simple filtering and statistical-based classification. Image segmentation is also performed, mainly to distinguish between features, surface, and background. Similar techniques can also be used for the identification of plumes and jets from comets and moons as done

*PhD Student, Department of Aerospace Science and Technology, Politecnico di Milano, mattia.pugliatti@polimi.it.

†PhD Student, Department of Aerospace Science and Technology, Politecnico di Milano, michele.maestrini@polimi.it.

‡Assistant Professor, Department of Aerospace Science and Technology, Politecnico di Milano, pierluigi.dilizia@polimi.it.

§Associate professor, Department of Aerospace Science and Technology, Politecnico di Milano, francesco.toppoto@polimi.it.

in [3]. Bodies are modeled using convex-hulls and pixels lying outside the hull are classified based on pixel intensity as belonging to the body's surface or a plume. The success of the technique is based on the difference between the sharp edges of the body and the more diffuse edges of the plume, which shall not be able to trigger an edge detection algorithm. With the advancement of Artificial Intelligence and deep-learning architectures, applications for image segmentation have also being developed for various applications. In [4] for example, a new successful architecture is introduced to perform image segmentation for biomedical application. The intuition is twofold: to adopt a symmetrical architecture that lacks the fully connected layer between encoding and decoding stacks and the idea to concatenate copy of trained encoder layers on the decoder to retain features information. In [5] a set of 5 CNNs are designed to detect geological structure on the surface of Mars at different scales. High-resolution images are taken by two cameras of the Mars Reconnaissance Orbiter and image segmentation is performed with CNN and compared with other methods. It is shown that CNN surpasses in accuracy other state of the art methods used for the same task-based on Support Vector Machine. This architecture is used in works such as in [6] and [7] to obtain hazard maps from the digital terrain map of the Moon. In these works, the U-Net is used to determine the horizontal coordinates of a safe landing spot on the lunar surface in an innovative way to allow a Deep-Reinforcing Learning method to perform a controlled landing on the surface. However, the aforementioned works do not fully appreciate the complexity of the features existing on a small-body surface, often not considering more than 3 layers. Also, an important trait they have in common is the lack of labeled data to be used for training. Most of the time this data is characterized manually from real images or derived from height-maps.

In this work, an approach is proposed for the automatic labeling of surface morphological features on small-bodies. This is exploited to generate a generalized sample of image-mask pairs that are used to train a CNN, enabling semantic segmentation for on-board applications. A similar approach has also been proposed in [8], where the 3D model of a satellite is rendered, together with automatic bounding boxes and ground truth pixel segmentation masks of its main features. The dataset is then used to train a network for the task of instance segmentation. The capability of assigning labels to pixels automatically unlocks new possibilities for on-board applications such as more complex hazard avoidance during landing, autonomous planning of scientific operations, and labeling of features of scientific interest as well as navigation purposes. The latter has been explored as a continuation of this work in [9], where a method based on CNN and Normalized Cross-Correlation is designed to work with small-body segmentation maps.

Blender, an open-source ray-tracing rendering software *, is used to generate high-fidelity images alongside their corresponding pixel masks thanks to ray-tracing techniques. An arbitrarily large annotated database is automatically generated and exploited for the training of a robust U-Net for semantic segmentation. The aim of automatically generating synthetic annotated images for this task is to extend the capabilities of the existing frameworks, to generalize for different models, and to better understand the complexity of the scene. To assess the performances of the trained network four different test cases are envisioned and discussed in this paper.

The paper is organized as follows. After presenting the automatic dataset generation pipeline in the next section, an overview of the CNN used to perform the semantic segmentation task is provided. Subsequently, the assessment of the results on the envisioned test cases is given in a dedicated section, which is followed by some final considerations and cues for future work that conclude the paper.

DATASET GENERATION

The small-body models used in this work are obtained by applying procedural changes of the mesh and improved textures to pre-existing low-poly asteroid models^{†‡}. The base models considered are: 67-P, Bennu, Didymos, Golevka, Hartley, HW1, Hygiea, Lutetia, and Thisbe.

After the mesh is modified to generate the desired level of roughness, craters are applied by extracting a

*<https://www.blender.org/>

†<https://sbn.psi.edu/pds/shape-models/>

‡<https://3d-asteroids.space/>

height-map from real craters on Earth*, an example for Barringer Crater can be seen in Figure 1.

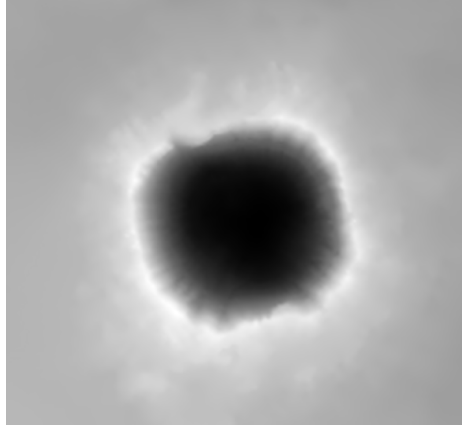


Figure 1. Grayscale heightmap of Barringer Crater used as texture for the craters on the small-body raw models.

A higher altitude is associated with brighter pixel values, as it can be seen from Figure 1. These images can be used in a texture to displace a planar object height, producing a realistic-looking crater. Applying random scaling, the generated craters are then stitched on the small-body models and distributed across its surface. Subsequently, boulders are generated and applied to the 3D models. Exploiting Blender’s Particle System tool, different families of randomly generated boulders are scattered on the surface. Each boulder has been created using the *Rock Generator* add-on in Blender. A summary of the craters and boulders number and size is provided in Table 1. After all morphological modifications have been put in place on the raw shape models, they are used to generate a database of images and masks. For each base shape, three different models are used to render image-mask pairs. The masks are constituted by 5 different layers (i.e. categories): background, free surface, craters, boulders, and terminator region. The masks are obtained in Blender by applying different object pass indexes and exploiting ray-tracing capabilities of the *Cycles* rendering engine. The three models used are referred to as:

- **Clean Model:** it is a simple model without surface texture, it is used to generate the ground truth of the terminator.
- **Crater Model:** it is a model of the asteroid with craters and textures applied, a boolean image with only visible craters can be produced. In these models, the craters’ mesh is not merged with the body.
- **Complete Model:** it is a model of the asteroid with textures, craters, and boulders, which is used to generate boolean images of the boulders and the free model surface, together with the full input image that will be used by the U-Net.

Once these three models are produced for each body, random camera positions are sampled uniformly in a spherical shell whose radius span $[0.4D_0, 1.3D_0]$, where D_0 is the approximate range at which the asteroid saturates the FOV of the camera, which is assumed to be 10 deg. Notice that an ideal pointing to the center of the small-body is assumed.

$$D_0 = \frac{\text{maximum size of asteroid}}{2 \tan(\text{FOV}/2)} \quad (1)$$

Moreover, for each acquisition, the sun’s direction is selected randomly in an angular range from the camera boresight of ± 90 deg. In such a way, off-nominal illumination conditions are taken into account without considering unrealistic image acquisition where the sun is behind the asteroid.

*<https://tangrams.github.io/heightmapper/>

Thanks to the 3 models, all ground truth layers but the terminator is obtained. For the latter, a dedicated processing pipeline is designed in Matlab*. First, a Canny edge extractor [10] is applied to the image generated from the Clean Model, which gives the edges detected in the image both between asteroid and space and those produced by the terminator. To avoid considering spurious edges (i.e. those between asteroid and space), an acceptance mask is computed which exploits the asteroid pass index mask which does not account for shadows. However, this binary image would not exclude the body-space edges obtained via the Canny extractor. To avoid including them, the boolean acceptance image is eroded using a morphological operation with circular structuring elements [11]. A summary of the process is graphically provided in Figure 2.

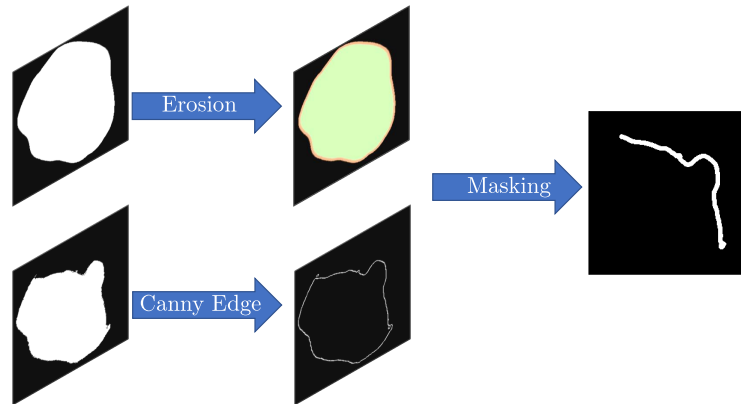


Figure 2. Extraction of terminator region from Clean Model.

Once all the raw masks are generated for every camera-sun pair, the stack of 5 masks is piled together, as in Figure 3. Note that the ground truth pixel masks obtained in this way may exhibit slight pixel intersection over categories, hence when the masks are stacked together a hierarchy is used to define precedence among layers. Intuitively, the hierarchy used in order of decreasing priority is: terminator, boulders, craters, surface, background. An example of this stacking process is reported in Figure 4.

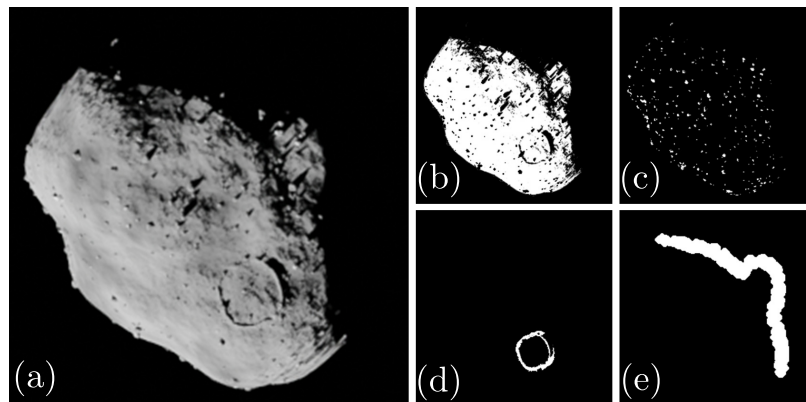


Figure 3. Complete image of the small-body model of Lutetia (a), together with its raw segmentation maps: (b) free surface, (c) boulders, (d) craters, (e) terminator.

The generated images are not directly used as input to the neural network. Indeed, they are further modified to account for noise and processing by adding a 0.1 mean 0.0001 variance Gaussian noise, followed by a 2D Gaussian smoothing kernel with a standard deviation of 0.1.

*<https://it.mathworks.com/products/matlab.html>

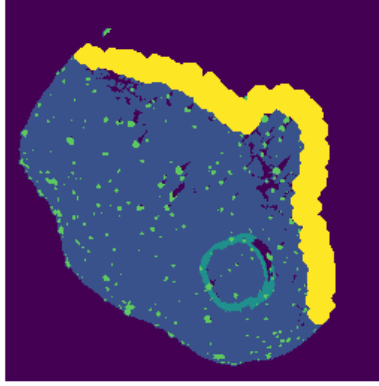


Figure 4. Segmented image stack constituting the truth mask to train the network.

This methodology of building models, rendering, and generating image-mask pairs allows to virtually produce an arbitrary amount of annotated samples, limited only by the available computational resources. Independently from the body shape, illumination conditions, features distributions, and scale it is then possible to use this database to efficiently train a U-Net in the task of image semantic segmentation for small-body morphological features.

Table 1. Summary of complete Blender models generated

Base Model	Craters no.	Boulders no.		
		<i>small</i>	<i>medium</i>	<i>large</i>
67-P	2	500	-	5
Bennu	3	1000	250	10
Didymos	4	800	30	5
Golevka	2	800	-	40
Hartley	3	5000	30	5
HW1	2	2000	40	5
Hygiea	5	1500	100	5
Lutetia	5	1000	350	-
Thisbe	5	1000	-	20

CONVOLUTIONAL NEURAL NETWORK FOR SEGMENTATION

The Network architecture selected for the segmentation task is inspired by the U-Net design [4]. This is motivated by its simplicity in the implementation in Tensorflow and because its efficacy has been already proven in segmentation tasks and space applications [6, 7, 4]. The network architecture used in this work is summarized graphically in Figure 5. The U-Net architecture is similar to the one of an autoencoder, but with tensors as input-output. The contracting portion of the network (the encoder) is composed of a succession of convolution, Rectified Linear Unit (ReLU) activation, and pooling layers of increasing depth and reduced size (i.e. height and width), while the expansive portion (the decoder) is made by a combination of transpose convolution, ReLu, and upsampling layers of reducing depth and increasing size. This symmetric nature is what gives the characteristic U shape. On top of this, the specific U-Net architecture introduced in [4] uses layers that are copied from the encoder and stacked in the decoder and lacks a fully connected layer in the middle separating encoding and decoding portions.

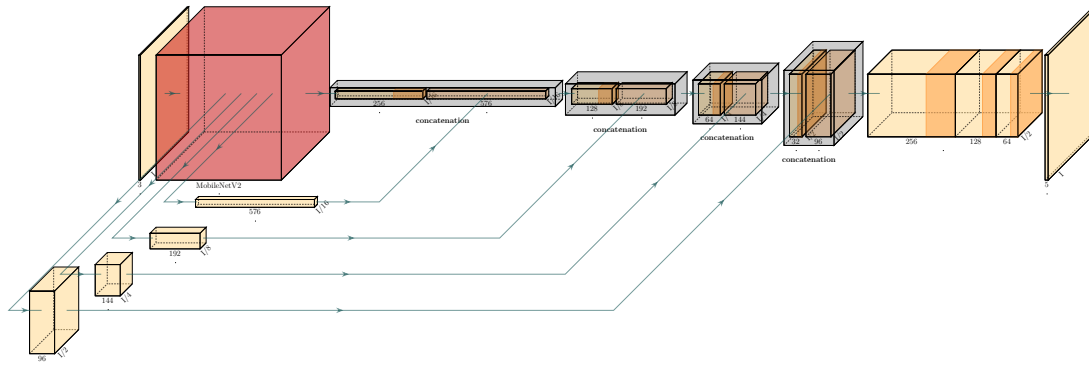


Figure 5. Custom U-net architecture. The input is the raw image, the output is the semantic segmentation of the content of the image.

For our particular case study, we implemented a custom architecture of the UNet. Our encoder follows the typical architecture of a CNN and is taken directly from a pre-trained model of MobileNet V2 [12] on the ImageNet dataset [13]. Its parameters are frozen during the training process to exploit the already tuned network and reduce the training time and computational cost. Indeed, the encoding branch of the network is responsible for extracting low-level features from the input image, which are often common for many problems from very different domains. In particular, we have extracted 5 activation maps from the early stages of the MobileNet network with dimension (height \times width \times depth): $4 \times 4 \times 320$, $8 \times 8 \times 576$, $16 \times 16 \times 192$, $32 \times 32 \times 144$, and $64 \times 64 \times 96$. These are then exploited in the decoding branch. As far as the decoder is concerned, it consists of 4 subsequent deconvolutions (or transpose convolution) stages. Each stage of the decoder starts with a deconvolution of the input with 3×3 kernels, which doubles the height and width of the input. Then, a ReLu activation is applied to the newly obtained feature map. The number of filters used in the decoder determines the depth of the feature map at each deconvolution step and it halves at each stage going from the initial 256, down to 32. In our network, the first deconvolution is applied to the last feature map extracted from the decoder (i.e. the $4 \times 4 \times 320$). The output of the first layer of the decoder is then concatenated to the encoder feature map of coherent height and width (i.e. $8 \times 8 \times 576$). In this manner, it is possible to exploit the additional features obtained during the encoding of the input image also for the decoding process to preserve higher accuracy during the upscaling. Afterward, to these concatenated feature maps, a 20% dropout is applied to provide regularization during training. The obtained stacked feature map is then used as the input to the next deconvolution, and the sequence (i.e. deconvolution, activation, concatenation, and dropout) is repeated for all 4 stages of the decoder. To conclude, the described layers of the network are followed by 3 additional convolutional layers obtained with 256, 128, and 64 filters of size 3×3 . Each layer is followed by a ReLu activation and 40% regularization dropout which grants more flexibility to the network. The output feature map of the network head is a $128 \times 128 \times 5$ feature map, where each of the 5 layers of depth stores a probability that the pixel belongs to a certain class. By applying a softmax across the depth of this feature map we can obtain the desired segmentation mask for the input image. The resulting network has a total of 4,450,757 parameters out of which only 2,607,813 are trainable.

Training

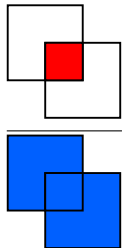
In order to train the network, first we need to define metrics showing how good the prediction of the network is when compared to the ground truth segmentation mask. One of the metrics used is the sparse categorical cross entropy. For each pixel \mathbf{x} we can create a softmax function $p(\mathbf{x})$ as:

$$p(\mathbf{x}) = \sum_{k=1}^K l(\mathbf{x}, k) \frac{\exp(a_k(\mathbf{x}))}{\sum_{k'=1}^K \exp(a_{k'}(\mathbf{x}))} \quad (2)$$

where $a_k(\mathbf{x})$ denotes the activation in the feature channel k at the given pixel \mathbf{x} and K is the number of classes (i.e. 5). The function $l(\mathbf{x}, k)$ is the delta Dirac function, which returns 1 if the pixel \mathbf{x} belongs to the category k , 0 otherwise. The loss will then be defined as the mean over all N pixels of the negative logarithm of $p(\mathbf{x})$

$$E = -\frac{1}{N} \sum_{\mathbf{x}} \log(p(\mathbf{x})) \quad (3)$$

To assess the accuracy of the trained CNN, we introduce two additional metrics that are used during the remainder of this work. First, we define global accuracy as the percentage of correctly classified pixels in a predicted segmentation map. Secondly, we define the mean Intersection over Union (mIoU), which is particularly relevant for the case of segmentation and can be computed for each category as a simple ratio between two areas. The numerator of this ratio is the area of overlap between the predicted pixel mask and ground truth pixel mask. On the other hand, the denominator is the total area encompassed by these two masks. Hence, for each pixel class, we can compute an IoU as

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{\text{Area of Red Square}}{\text{Area of Blue Squares}} \quad (4)$$


The network is trained only once for all the following test cases. In particular, 7 arbitrarily selected models have been chosen among the available ones to be used during the training of the network: 67-P, Bennu, Didymos, Golevka, Hartley, Hygiea, and Lutetia. For each model, 1500 images are generated as described in the dedicated Dataset Generation section. The preliminary dataset showed a shortcoming in the number of craters that appeared during training. Hence, an additional 3,100 images of the craters on the 7 asteroid models have been added to the training set. A summary of the dataset is reported in Table 2, together with its split among training set and validation set.

Table 2. Summary of the synthetic image-mask pairs constituting the training, validation and test set.

Base Model	Training		Validation Set
	<i>Database</i>	<i>Additional Craters</i>	
67-P	1200	150	150
Bennu	1200	650	150
Didymos	1200	550	150
Golevka	1200	300	150
Hartley	1200	300	150
Hygiea	1200	650	150
Lutetia	1200	500	150
Tot.		11500	1050

The training parameters for the network have been identified after a large hyperparameter search. The selected batch size is 100 images, and the network was trained for 300 epochs with 115 gradient steps per

epoch. The selected optimizer was Adam [14], with a learning rate of 5×10^{-4} . The curves describing the training process in terms of mIoU, accuracy, and loss function are represented in Figure 6.

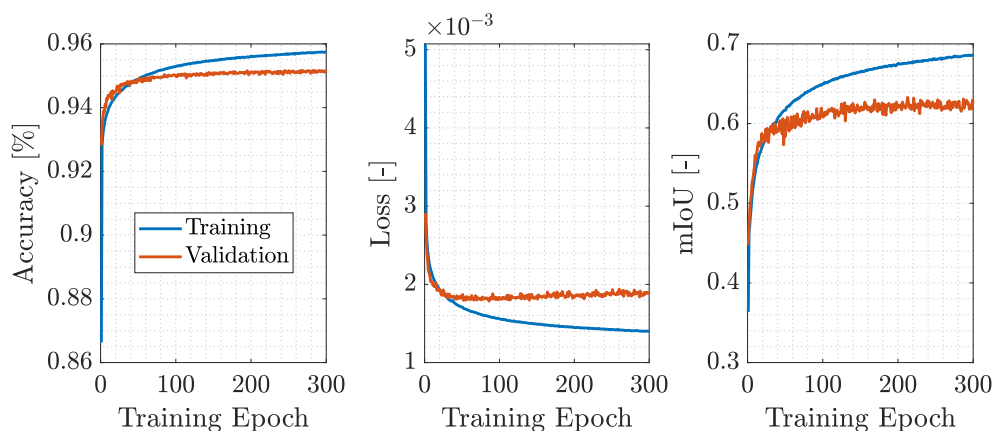


Figure 6. Evolution of the values of accuracy (left), loss (center), and mIoU (right) obtained during the training process. Both curves for the training (blue) and validation (orange) sets are reported.

RESULTS

The trained CNN undergoes 4 test cases. Each of these tests has a dedicated dataset which is better explained in Table 3. A summary of all the results of performance in terms of both Accuracy and mIoU is reported in Table 4 and Table 5. Notice that no manual labeling has been performed for D-4. Therefore, the performance will be assessed qualitatively.

Table 3. Summary of the datasets used in the four test cases, identified as *D-* followed by the test identification number.

Dataset	Models	Used in Training	Number of Images
D-1	7 (67-P, Benu, Didymos, Golevka, Hartely, Hygiea, and Lutetia)	✓	1050
D-2	2 (HW1, and Thisbe)	×	3000
D-3	1 (Thisbe)	×	56
D-4	/	×	59

Table 4. Table with a summary of accuracy for the different test cases expressed as a percentage.

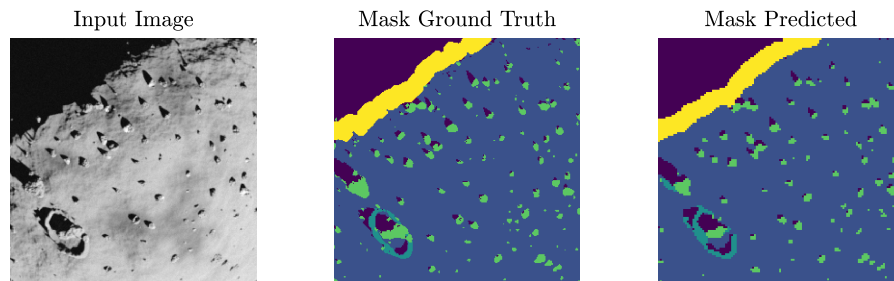
Test case	D-1			D-2			D-3			D-4		
	max	min	avg	max	min	avg	max	min	avg	max	min	avg
Background	100.00	0.00	92.32	100.0	0.00	96.35	100.00	56.25	97.99	/	/	/
Surface	99.83	80.17	97.73	99.83	86.41	98.28	100.00	49.25	96.16	/	/	/
Craters	100.00	0.00	37.68	100.00	0.00	19.69	100.00	0.00	8.07	/	/	/
Boulders	95.06	0.00	31.37	79.78	0.00	25.22	100.00	0.00	15.14	/	/	/
Terminator	100.00	0.00	80.71	100.00	0.00	66.23	100.00	0.00	65.36	/	/	/
Mean		95.03			95.09			97.65		/	/	/

Table 5. Table with a summary of IoU for the different test cases expressed as a percentage.

Test case	D-1			D-2			D-3			D-4		
	max	min	avg	max	min	avg	max	min	avg	max	min	avg
Background	100.0	10.00	89.35	100.00	7.69	92.82	100.00	50.00	95.99	/		
Surface	99.67	71.39	92.33	99.51	44.78	91.92	99.13	36.56	82.93	/		
Craters	100.00	0.11	28.31	100.00	0.10	17.00	100.00	0.57	15.58	/		
Boulders	86.07	0.51	27.83	69.73	0.78	21.39	58.73	1.79	18.63	/		
Terminator	100.00	0.61	66.67	100.00	0.15	58.24	100.00	0.61	58.67	/		
Mean	60.90			56.28			43.95			/		

Test case 1

In this first test, the test dataset D-1 from Table 3 has been used. This test case is standard for CNNs and consists of evaluating the performance of the CNN on a split of the dataset which was not used during training. Hence, the base models which are included in this dataset are the same that were used to generate the training dataset. As expected, the network achieves an accuracy of $\approx 95\%$, which is consistent with the validation data observed during training. The mIoU on this dataset is $\approx 61\%$, which testifies a good performance of segmentation. As an example, we report one of the best (Figure 7) and one of the worst (Figure 8) performances obtained, a mosaic showing an overview of some images, predictions, and ground truth is also provided in Appendix. We can already observe that the accuracy has little meaning in terms of segmentation accuracy and that mIoU constitutes a far better metric. An additional issue which is evidenced in Figure 8 and can be observed through the entire dataset, is that when moving far away from the surface, the boulders and craters become less easy to spot, and the network fails to identify them. Indeed, as can be observed from the cumulative performance curves of Figure 9, the categories of *Boulders* and *Craters* are responsible for dragging the performance of the network down. The reason for this is due to the statistics of the training set, where the categories best represented in terms of percentage of total pixel datasets are in descending order: surface 52.35%, background 38.93%, and terminator 4.41%. Instead, the least represented categories are the boulders with 0.88% and craters with (even after the injection of additional data) 2.43%. Despite this justification, the observed performance drop cannot simply be explained by the pixel statistics. Indeed, we must also consider that contrarily to the easier categories (background, face, terminator), the possibility to identify craters and boulders strongly correlates with their size as well as the asteroid phase angle. Therefore, these features are intrinsically harder to detect.

**Figure 7. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 95\%$ and a mIoU of $\approx 74\%$.**

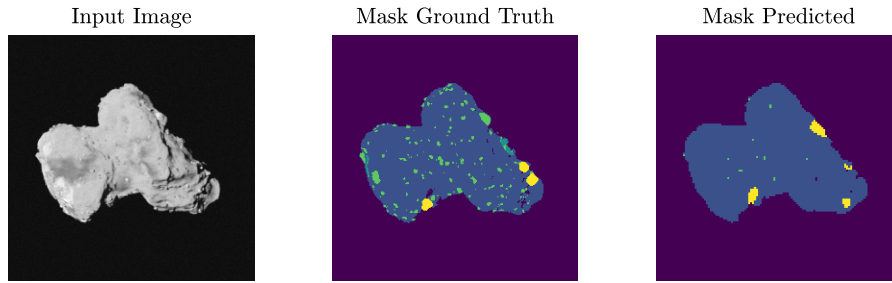


Figure 8. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 96\%$ and a mIoU of $\approx 41\%$.

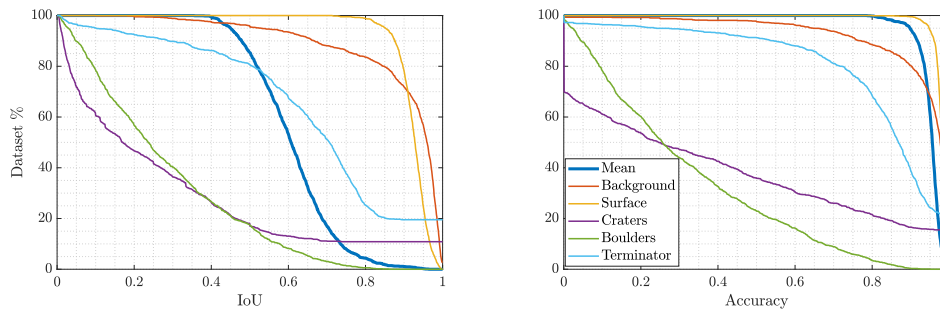


Figure 9. Cumulative performance curves indicating which percentage of the dataset lies above a certain value of IoU (left) or accuracy (right) for dataset D-1.

Test Case 2

In this second test, the test dataset D-2 from Table 3 has been used. This test case is meant to determine if the trained CNN is capable of generalizing the learned features across different (unknown) small-bodies. Indeed, this second dataset consists of two artificial models that were not used to generate the training images. The network achieves an accuracy of $\approx 95\%$, which is consistent with the performance of D-1, whereas the mIoU drops to $\approx 56\%$, which still testifies that a good segmentation is retained. Therefore, we can assume that the network did not overfit on the training models, and is capable of partial generalization on similar synthetically generated data. As an example, we report one of the best (Figure 10) and one of the worst (Figure 11) performances obtained, however a larger mosaic providing an overview of some images, predictions, and ground truth is reported in Appendix. We can already observe a similar behavior to Test Case 1, with a slight reduction in mIoU, which is to be expected given the novelty of the models. This second test also highlights the same shortcomings of Test Case 1 in terms of the capability to identify the boulders due to high scale variance. Indeed, also in this case it can be observed that craters and boulders are responsible for dragging down the overall score (e.g. Figure 12).

Test Case 3

In this third test, the test dataset D-3 from Table 3 has been used. This test case is meant to provide some insight into the behavior of the network during a fly-by trajectory. The spacecraft keeps an ideal pointing towards the asteroid while passing by it. Most of the dataset ($\approx 70\%$) has been taken on a range of distances

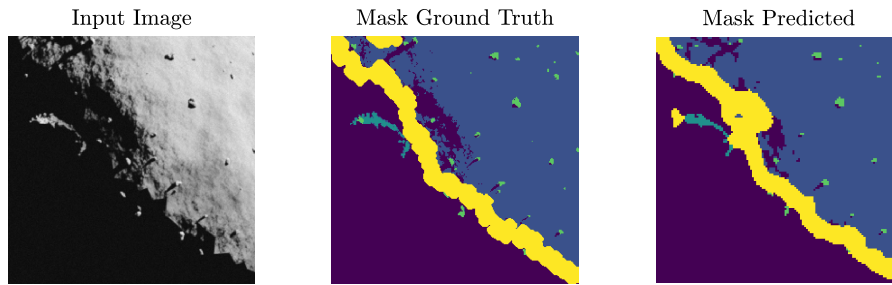


Figure 10. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 94\%$ and a mIoU of $\approx 72\%$.

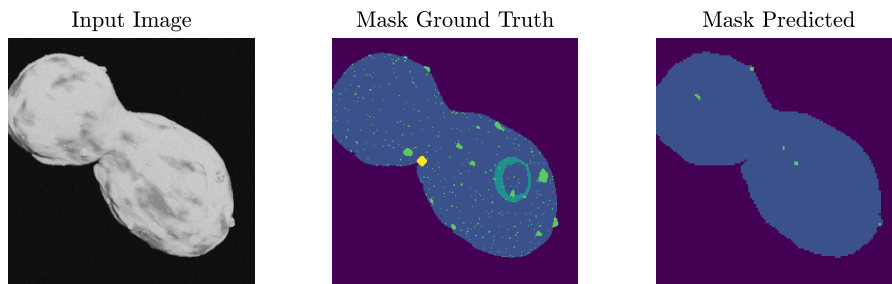


Figure 11. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 97\%$ and a mIoU of $\approx 40\%$.

outside the training range of $[0.4D_0, 1.3D_0]$, and this is reflected on the mIoU which is the lowest seen up until this point (see Table 5). The behavior of the mIoU during the trajectory is reported together with the range from the asteroid in Figure 13. As expected, the best performance is retrieved in the range of distances for which the network has been trained, however, the performance drop outside the training range is not steep. This can also be observed in one of the best (Figure 14) and worst (Figure 15) performances on this dataset. Additionally, a mosaic showing an overview of some images, predictions, and ground truth is provided in Appendix.

Test Case 4

In the last test, the dataset D-4 from Table 3 has been used. This test case provides some insight into the applicability of this approach to real case scenarios. To the best of our knowledge, no other analysis of this kind is available in the literature. Despite the shortcoming of the proposed approach, this assessment provides precious insights to improve the approach in future studies. Given that the data are not labeled manually, no quantitative analysis has been performed, and only a qualitative assessment is obtained. The dataset contains two types of images, the first being 9 entire images which are downsampled to comply with the input shape requirement. On the other hand, the second larger part of the dataset (i.e. 50 images) consists of crops of

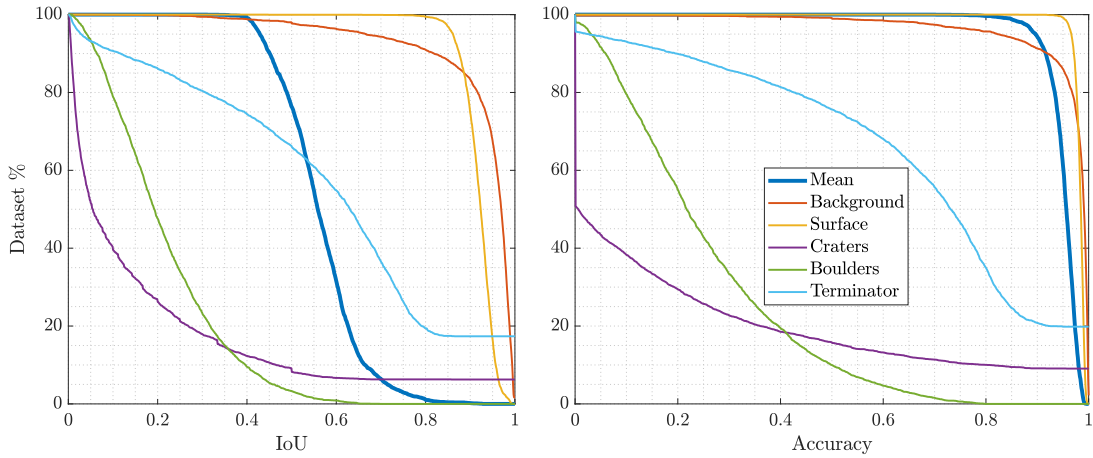


Figure 12. Cumulative performance curves indicating which percentage of the dataset lies above a certain value of IoU (left) or accuracy (right) for dataset D-2.

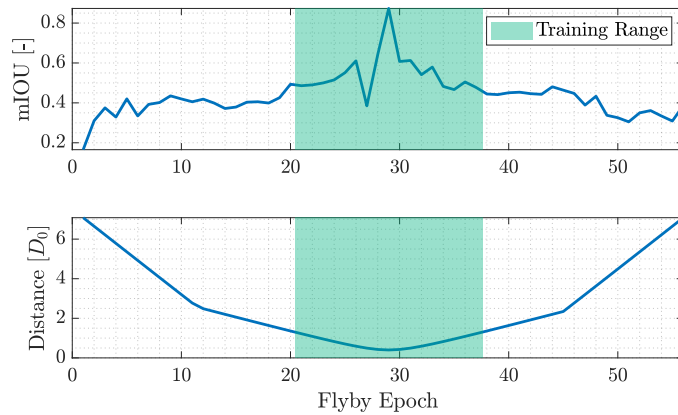


Figure 13. Evolution of the mIoU during the flyby. Notice that the best performance is retrieved in the range of distances for which the network is trained.

real images without any downsampling. The images include the following bodies: Itokawa^{*}, Eros[†], Vesta[†], Bennu[†], and Ida[†].

The results on the resized images presented very promising results, where the categories of terminator, surface, and background are well classified. This behavior can be observed in Figure 17 as well as the first 9 images of the mosaic provided in Appendix.

However, these images also underlined the complete lack of identification of craters and boulders. As for the first test cases, we decided to verify if the lack of identification of these categories depended on the relative size of these features in the images. Therefore, we decided to take zoomed crops of the original images and test our hypothesis. As observed from Figure 18, the major boulders are now identified sufficiently well. Despite this improvement, we can still observe several shortcomings and mislabelling when the network is applied to real images, which are evidenced in the mosaic reported in Appendix. For completeness, we also report one of these failures in Figure 19, where the lack of precision in identifying the main crater and the

^{*}<http://jda.jaxa.jp/en/>

[†]<https://photojournal.jpl.nasa.gov/>

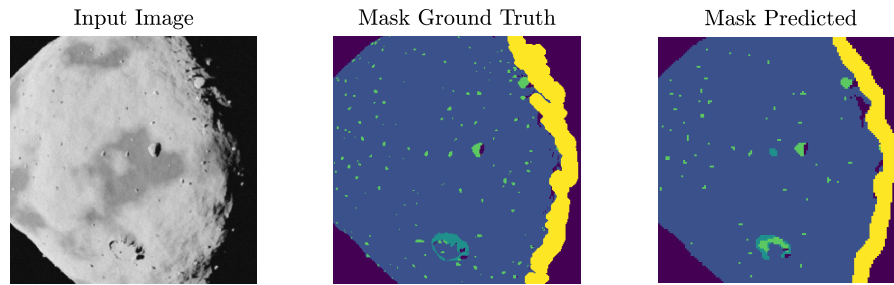


Figure 14. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 96\%$ and a mIoU of $\approx 58\%$.

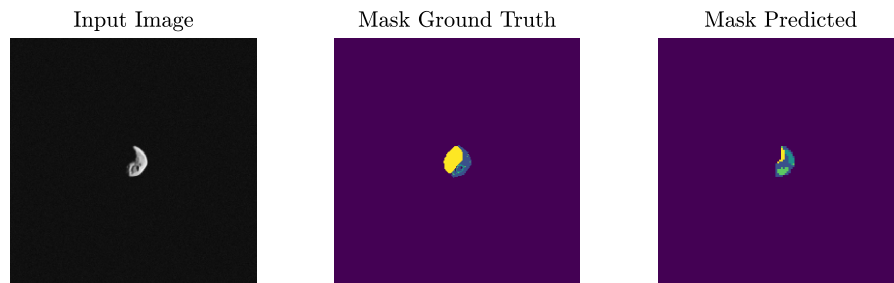


Figure 15. Triplet of images showing input image, ground truth and predicted mask. The image in question shows an accuracy of $\approx 99\%$ and a mIoU of $\approx 17\%$.

confusion between craters and boulders appears evident. This analysis allowed us to determine that the crater modeling is not realistic enough for real image application and requires more work. Besides, we deem that the training could improve by augmenting our synthetic dataset with some real, manually labeled images.

CONCLUSION AND FUTURE WORKS

In this work, an approach is proposed for the automatic labeling of surface morphological features. The methodology presented exploit Blender's capabilities and ray-tracing techniques to extract layers such as background, surface, craters, boulders, and terminator region. This capability is used to generate a database of image-mask pairs that are used to train a CNN for on-board semantic segmentation. The CNN has been designed using a U-Net inspired architecture. Accuracy and mIoU have been used as metrics to assess the CNN's performances, which have been tested on 4 different test cases. The CNN demonstrated to be capable to segment the images with a high value of accuracy, but more importantly with a good value of mIoU. The CNN also demonstrated to be capable to generalize its performances on models that have never been seen during training, with only a moderate drop in performances. The CNN also demonstrated that when applied out of the range considered in the training, it performed poorly in a flyby scenario, while it performs as expected within the training range. When tested with synthetic images the CNN has demonstrated to have issues at detecting craters and boulders from higher distances. This is something that can be resolved in

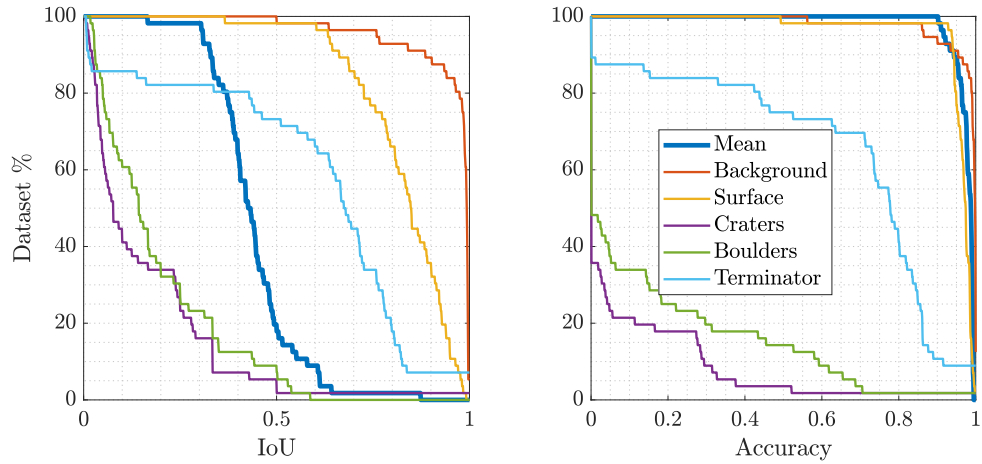


Figure 16. Cumulative performance curves indicating which percentage of the dataset lies above a certain value of IoU (left) or accuracy (right) for dataset D-3.



Figure 17. Predicted mask for an image of Bennu.

future developments with adoptions of dynamic class-based weights factors. The performances of the CNN also fall short in the segmentation of real images from previously flown missions. surface, background, and terminator (that are generally easier to detect) are determined confidently, the same cannot be said about craters and boulders. This could be attributed to a wrong representation of these features in the synthetic images in contrast with the real ones. Future works are envisioned towards more accurate modeling of craters, on the injection of real labeled images in the training dataset, and in the usage of weight, factors to be used for training the CNN with particular attention to the most complex features.

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. The authors would like to thank Carmine Buonagura for his work on the procedural asteroid generator tool, which formed the base of the software used in this work to generate raw shape models.

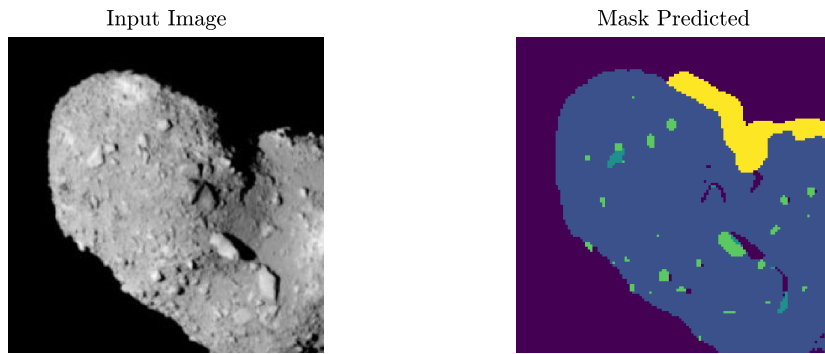


Figure 18. Predicted mask for an image of Itokawa.

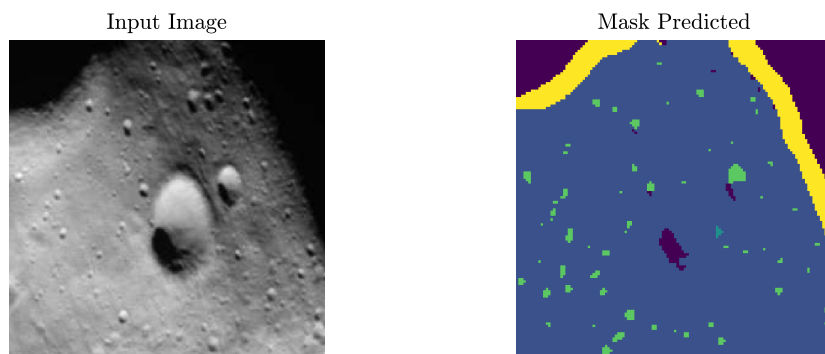


Figure 19. Predicted mask for an image of Ida.

APPENDIX: MOSAIC

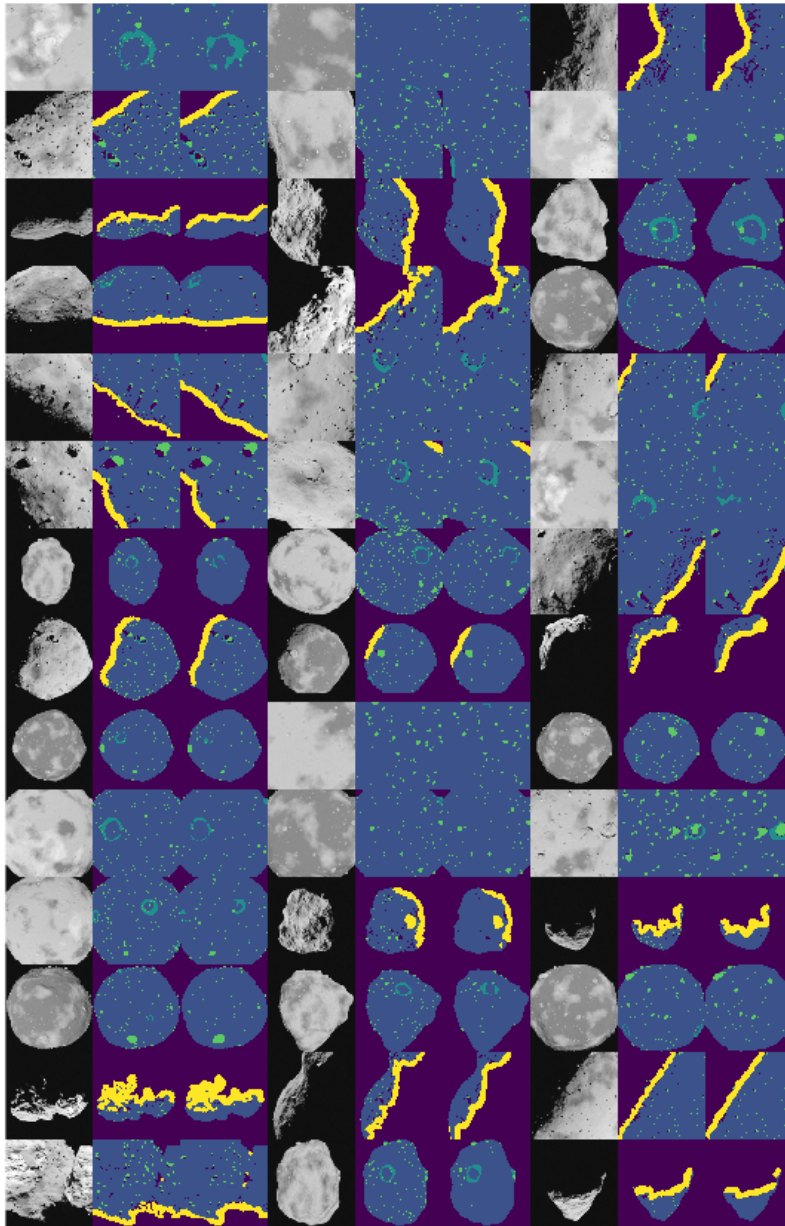


Figure 20. Mosaic of 3×14 triplets showing input image, ground truth and predicted mask for test D-1. Each triplet is composed of the input image (left), ground truth segmentation mask (center) and predicted segmentation mask (right).

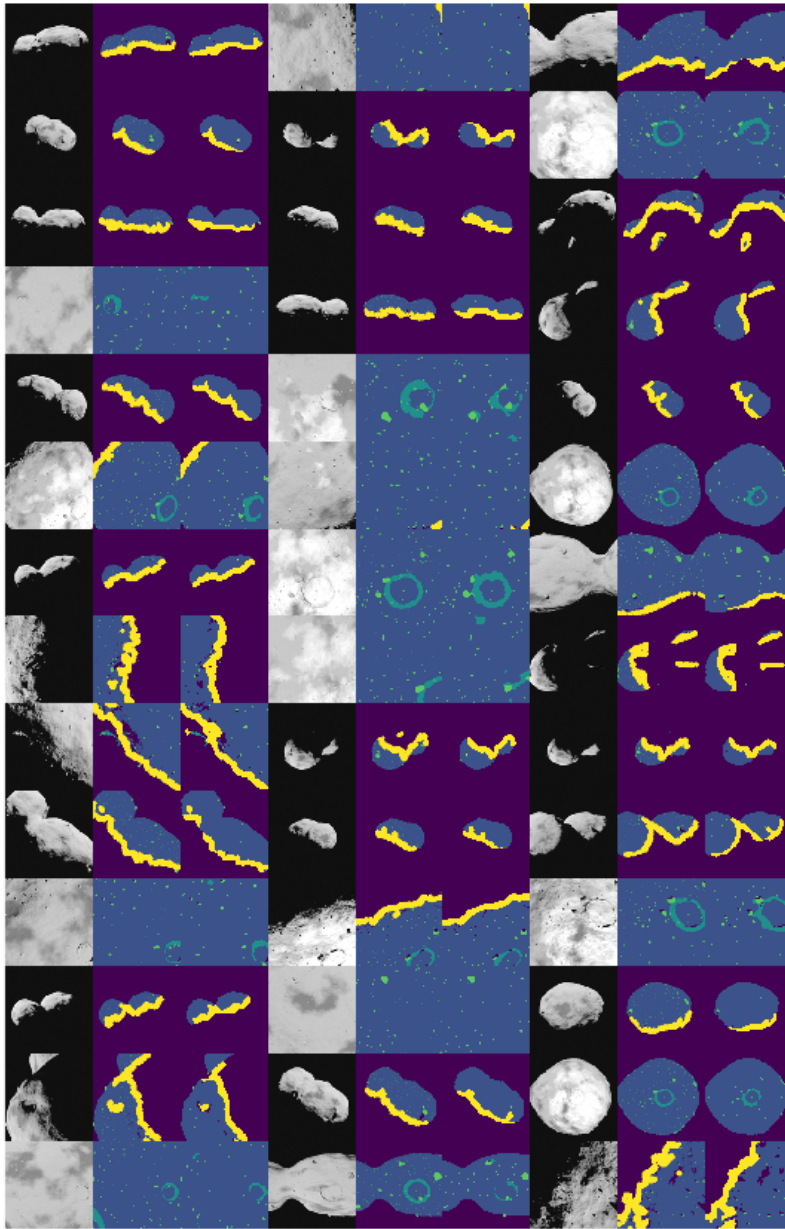


Figure 21. Mosaic of 3×14 triplets showing input image, ground truth and predicted mask for test D-2. Each triplet is composed of the input image (left), ground truth segmentation mask (center) and predicted segmentation mask (right).

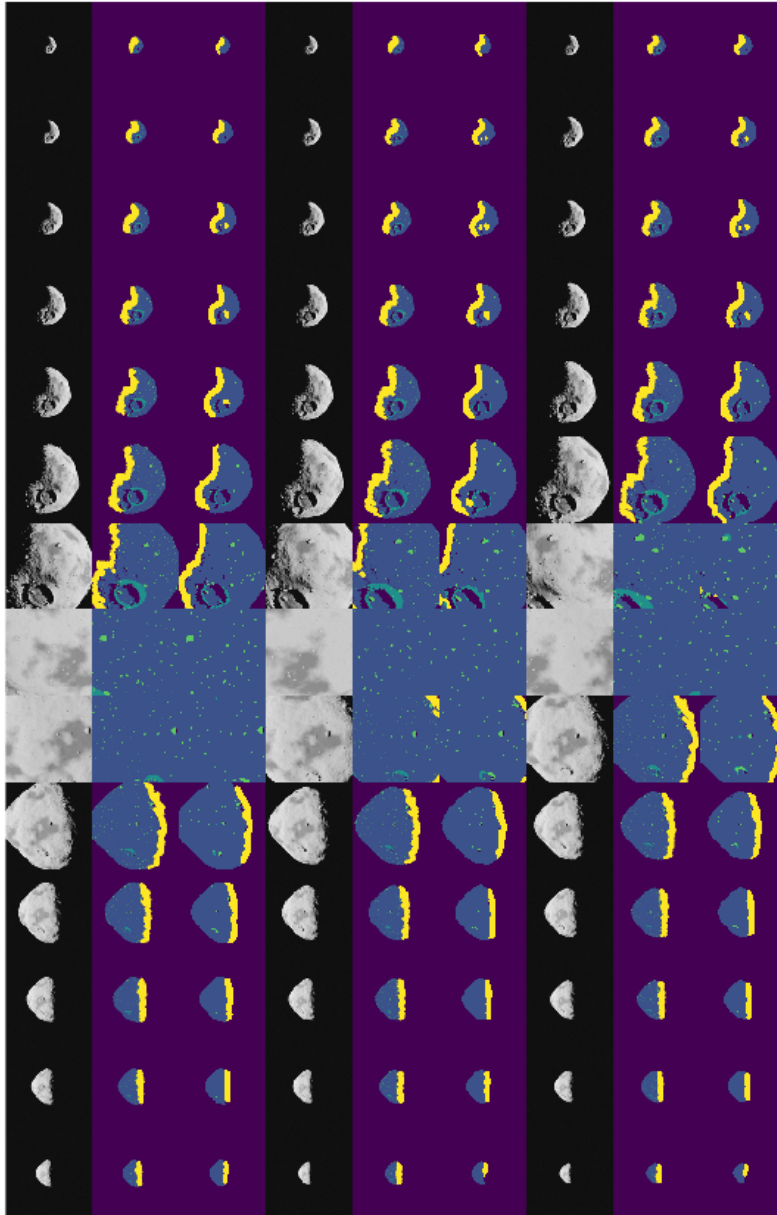


Figure 22. Mosaic of 3×14 triplets showing input image, ground truth and predicted mask for test D-3. Each triplet is composed of the input image (left), ground truth segmentation mask (center) and predicted segmentation mask (right).

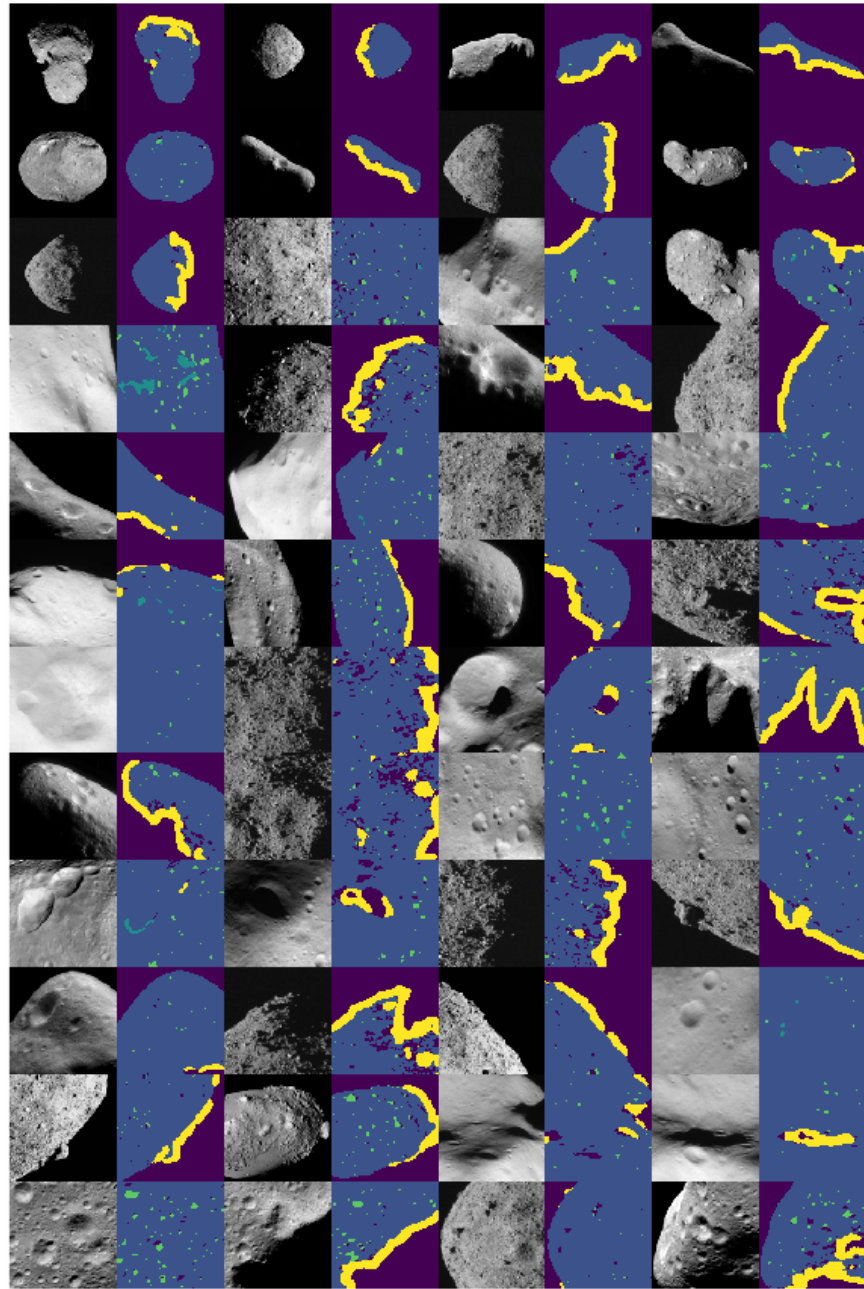


Figure 23. Mosaic of 3×12 couples of images showing input image and predicted mask for test D-4. Each couple is composed of the input image (left), and predicted segmentation mask (right).

REFERENCES

- [1] D. Thompson, S. Niekum, T. Smith, and D. Wettergreen, "Automatic detection and classification of features of geologic interest," *Proceedings. of IEEE Aerospace Conference*, 2005, pp. 366–377, <https://doi.org/10.1109/AERO.2005.1559329>.
- [2] 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, Oct. 2015, pp. 417–434, <https://doi.org/10.1002/2014ea000042>.
- [3] 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, Oct. 2014, p. 43, <https://doi.org/10.1088/0004-637X/794/1/43>.
- [4] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Vol. 9351 of *LNCS*, 2015, pp. 234–241, https://doi.org/10.1007/978-3-319-24574-4_28.
- [5] L. F. Palafox, C. W. Hamilton, S. P. Scheidt, and A. M. Alvarez, "Automated detection of geological landforms on Mars using Convolutional Neural Networks," *Computers & Geosciences*, Vol. 101, Apr. 2017, pp. 48–56, <https://doi.org/10.1016/j.cageo.2016.12.015>.
- [6] 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," *2020 AAS/AIAA Astrodynamics Specialist Conference, South Lake Tahoe, CA*, No. AAS 20-522, Aug 2020. (not published).
- [7] K. Iiyama, K. Tomitay, T. N. Bhavi A. Jagatiaz and, and K. Ho, "Deep Reinforcement Learning for safe landing site selection with concurrent consideration of divert maneuvers," *2020 AAS/AIAA Astrodynamics Specialist Conference, South Lake Tahoe, CA*, No. AAS 20-583, Aug 2020. (not published).
- [8] N. Faraco, "Instance segmentation for features recognition on non-cooperative resident space objects," Master's thesis, Polytechnic University of Milan, Department of Aerospace Science and Technology, Milan, Italy, 2020, <http://hdl.handle.net/10589/167418>.
- [9] M. Pugliatti and F. Topputo, "Navigation about irregular bodies through segmentation maps," *31st Space Flight Mechanics Meeting, Charlotte, NC*, No. AAS 21-383, Feb 2021. (not published).
- [10] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6, 1986, pp. 679–698, <https://doi.org/10.1109/TPAMI.1986.4767851>.
- [11] F. Shih, *Image Processing and Mathematical Morphology: Fundamentals and Applications*, ch. 3, pp. 25–35. CRC Press, 2011.
- [12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520, <https://doi.org/10.1109/CVPR.2018.00474>.
- [13] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255, <https://doi.org/10.1109/CVPR.2009.5206848>.
- [14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.