POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Noise-augmented Synthetic Training Dataset for Small Body images semantic segmentation

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Giovanni Saponaro, 103005

Advisor:
Prof.Michèle Roberta Lavagna

Academic year:
2024-2025

Abstract: The exploration of small bodies, such as asteroids and comets, requires increased spacecraft autonomy for critical operations like landing and Hazard Detection and Havoidance (HDA), where image-based semantic segmentation plays a key role. The major challenges in this field arise from the scarcity of labeled mission imagery and the domain gap between synthetic and real training data. This thesis addresses this challenges by proposing an improved state of the art procedural framework for the generation of synthetic datasets with automatic ground-truth labeling. Noise augmentation strategies are investigated to mitigate the domain shift, and two architectures, a baseline U-Net and a more advanced MultiRes U-Net, are trained exclusevely on synthetic data and tested on manually labeled images. Results show alignment with prior work by M.Pugliatti and M.Maestrini [19], with a marked improvement in background segmentation and an increase in the overall Mean Intersection over Union (MIoU). MultiRes U-Net achieved better results than the baseline, demonstrating superior generalization under domain shift. This study highlights the importance of training dataset design and the need to exploit more complex architectures and training strategy to further bridge the domain gap between synthetic and real data in support of autonomous space exploration.

Key-words: Small Body, Semantic Segmentation, Synthetic Dataset, Noise Augmentation, Deep Learning

Acronyms

ALHAT	Autonomous Landing and Hazard Avoidance Technology.
ANN	Artificial Neural Networks.
BRDF	Bidirectional Reflectance Distribution Function.
CNN	Convolutional Neural Networks.
DEM	Digital Elevation Model.
GNC	Guidance Navigation and Control.
HDA	Hazard Detection and Avoidance.
IoU	Intersection over Union.
LR	Learning Rate.
LRO	Lunar Reconnaissance Orbiter.
MIoU	Mean Intersection over Union.
NEO	Near Earth Objects.
OSL	Open Shading Language.
PBR	Physically Based Rendering.
WD	Weight Decay.

1. Introduction

The exploration of small bodies, such as asteroids, comets, and small natural satellites, is a rapidly expanding area within the space sector. The large interest in these targets is related to both their accessibility and scientific value. As primordial remnants of the solar system, small bodies offer invaluable insights into its origins and formation; furthermore, they are perfect targets for advancing in planetary defense, in situ resource utilization, and the ongoing quest for extraterrestrial life.

Small bodies in our solar system include Near Earth Objects (NEO)s, main-belt asteroids and comets, centaurs, and trans-Neptunium objects such as Kuiper Belt objects. Over 850000 small bodies have been observed in the solar system, but only 25 have been flown by to date, and a few have been rendezvoused with. NEAR Shoemaker (NASA, 1996–2001) [18], which orbited and landed on asteroid 433 Eros, Hayabusa (JAXA, 2003–2010) [27] and Hayabusa2 (JAXA, 2014–2020) [24], targeting asteroids Itokawa and Ryugu, OSIRIS-REx (NASA, 2016–2023) [11], which collected and returned samples from asteroid Bennu, and ROSETTA (ESA, 2004-2016) [26], which landed on 67P/Churyumov Gerasimenko comet, are some of the flown missions.

These missions have demonstrated the technical feasibility and scientific value of small body exploration. However, they also highlighted key challenges, such as uncertain terrain, limited prior knowledge of surface morphology, and the need for real-time decision-making with minimal Earth intervention.

As the distance from the Earth grows, the signal delay increases too, and real-time control cannot be applied during critical operations, such as landing, sample collection or unexpected scenarios, which are more common in almost unknown environments like small bodies. Moreover, ground operations costs can constrain both mission duration and the volume of scientific data collected.

In this context, where a-priori knowledge is limited or highly uncertain, autonomy becomes critical, as the spacecraft must be able to operate independently of external control. Spacecraft autonomy allows to navigate

in complex and unseen environments while adapting to unexpected scenarios without ground intervention, thereby enhancing mission flexibility and potentially reducing overall costs.
To achieve such autonomy, however, a clear understanding of the surrounding environment is required, and, for this purpose, cameras are often the preferred sensor. Cameras, alongside their lightweight, small size and low power consumption, can be exploited in different mission phases, from long distances to detect and characterize celestial bodies during flybys, to close range for descent and landing. In response to these needs, lot of image processing algorithm have been developed in the framework of autonomous navigation, but in the last decade artificial neural network have demonstrated to be able to out-perform state of the art techniques both in accuracy and computational resources demand.

1.1. Background

Autonomous landing on celestial bodies is one of the most critical phases of a space mission and it requires segmenting the surface into safe vs hazardous areas under tight timing and navigation uncertainty. The selection of a safe landing site is complex, since in many cases scientifically relevant areas coincide with hazardous terrain or are confined to small zones.
NASA's Autonomous Landing and Hazard Avoidance Technology (ALHAT) program [4] established the canonical baseline, with the aim of providing an autonomous Guidance Navigation and Control (GNC) system capable of landing on unseen terrain. This technology relies on the use of the *Flash LIDAR*, together with IMU, star tracker, altimeter and doppler velocimeter, through which the system reconstructs a Digital Elevation Model (DEM) of the terrain and characterizes the safety of the surface in a probabilistic framework in order to determine the safest landing locations.
Early attempts to move beyond purely sensor-based approaches include the work of Lunghi et al., who introduced an Artificial Neural Networks (ANN)-based Hazard Detection and Avoidance (HDA) system using monocular camera images and extracting local descriptor (variance, gradients, Laplacian of Gaussian) as inputs to a multilayer perceptron network to build a hazard map [10]. This work showed the potential of machine learning for real-time HDA, with efficient computation and adaptability to unseen hazards.
The breakthrough came in the last decade with the advent of Convolutional Neural Networks (CNN)s, which reframed the HDA problem as a pixel-wise classification problem, known as semantic segmentation.
Two families of approaches emerged: Lidar-based and camera-based.
The Lidar-based approach extends the ALHAT baseline by applying CNNs to DEMs.
Moghe and Zanetti [13] employed UNet-like CNN trained on DEMs from the Lunar Reconnaissance Orbiter (LRO) for slope estimation. The CNN produced dense probability maps of safe sites with high pixel accuracy and strong real-time suitability; it was able to achieve less than 1% False Safe location percentage thanks to the right input weights in the weighted Jaccard loss function.
Tomita et al. [23] showed that CNNs trained on synthetic DEMs could even outperform ALHAT baseline both in speed and accuracy. U-Net, SegNET and ICNet were compared both in binary and multi-class segmentation with discretized slope and roughness masks, highlighting how there is no clear advantage in predicting binary maps although multi-class segmentation has higher inference time.
However, Lidar-based methods rely heavily on the availability of DEMs, which can be reconstructed for large planetary bodies but are far less reliable in the case of small, irregularly shaped bodies. For these environments, where topography is complex, camera-based approaches become essential. Here the main bottleneck is not network design, but data: unlike terrestrial vision tasks, for which large labeled datasets are already available (ImageNet, COCO, Cityscapes), planetary exploration provides only limited imagery per mission, and manual pixel-level annotation is very labor-intensive. Data scarcity and the absence of annotated images thus stand as the central challenge.
It is precisely this challenge that M.Pugliatti and M.Maestrini [19] address. Their work significantly contributed to the state of the art in small body segmentation for safe landing through a comprehensive methodology that leverages U-Net architectures for semantic segmentation, classifying image pixels into morphological features such as background, surface, craters, boulders, and terminator region. Their approach crucially addresses data scarcity by developing an innovative framework for generating synthetic, automatically labeled datasets in Blender. This process involves 3D models of celestial bodies with simulated geological features and it automatically derives detailed segmentation masks, which are then augmented with Gaussian noise to enhance network robustness. This synthetic dataset is then integrated with a smaller, manually labeled dataset of real images providing a more realistic basis for training.

1.2. Scope of the study

The scope of this work is to investigate and improve methodologies for semantic segmentation of small body imagery using synthetic and real images. In particular, the study addresses the challenges of data scarcity and domain shift between synthetic and real mission imagery by proposing an enhanced data generation pipeline, exploring noise modeling strategies, and evaluating both baseline and advanced convolutional neural network architectures.
An improved framework for the procedural generation of automatically labeled datasets is proposed. The concept of this framework is inspired by the work of Pugliatti and Maestrini [19], but different tools are employed in order to improve, speed up, and simplify the generation of paired image-mask data. In addition, a noise analysis on real mission imagery is conducted to build an equivalent noise model, with the aim of enhancing segmentation performance when transferring from synthetic to real images. Using this framework, a large synthetic dataset is generated and a U-Net model is trained with different noise augmentation strategies and subsequently evaluated on a dataset of manually labeled real images. Finally, the MultiResidual U-Net architecture is introduced to assess whether more advanced network designs can achieve better generalization when trained exclusively on synthetic data under domain shift conditions.
Since this study and the work done by M.Pugliatti and M.Maestrini [19] share a similar segmentation task, a comparison can be done in order to assess whether the proposed framework improvements and noise augmentation strategies lead to measurable gains in segmentation accuracy on real data. However, the differences in training images and network architectures need to be taken in account during the evaluation, as they prevent a strict one-to-one comparison.
The study has three main objectives:

- Develop a more efficient and flexible procedural framework for producing synthetic datasets for small body with automatic ground-truth labeling, improving the previous work [19] in terms of speed, usability, and scalability.
- Improve the ability of networks trained on synthetic data to segment real imagery, by employing U-Net as a baseline with the adoption of a classical and a real-case-scenario noise augmentation strategies, and by comparing the results with those obtained by Pugliatti and Maestrini [19].
- Assess the performance of a more complex U-Net-based model, MultiRes U-Net, compared to the baseline, in order to verify its robustness and generalization capability under domain shift conditions.

The article is organized as follows. In Section 2 the Dataset Generation Framework is described, together with the description of the training datasets assembly. The Section 3 presents the two noise augmentation strategies adopted, followed by the characterization of the Artificial Neural Networks employed in this work and an overview on the training setup in Section 4. Subsequently, the assessment of the results on the proposed test cases is given in Section 5, concluding with final considerations and future works in Section 6.

2. Dataset Generation Framework

Ray-tracing constitutes a key technique when dealing with the generation of synthetic imagery and it is employed by many softwares available on the market. Planet and Asteroid Natural Science Generation Utility (PANGU) [15], developed by the European Space Agency (ESA), is designed to produce planet and asteroid natural scenes with high degree of realism. It allows to generate visible, thermal infrared, LiDAR and RADAR images from any position and orientation. On the other hand, Unreal Engine, Persistence Of Vision Ray-tracer (POV-ray) [16] and Blender are the most prominent solutions among the free and open-source tools. For the development of this thesis, Blender has been chosen as rendering software for its versatility, the availability of multiple rendering engines and the integration of Python; moreover it benefits of a wide user community.

2.1. 3D model

The adopted approach is based on the artificial enrichment of an already existing or synthetically generated small body shape model with morphological features.

2.1.1 Mesh

The first step consists of importing the simplest shape model of the target body (or generating a new one from scratch). At this stage, it is important to ensure that relevant morphological features are not visible, so that the model captures only the essential geometry.
The low resolution meshes are enhanced to generate a high resolution one up to a few million faces, by using the Subdivision Surface Modifier.

The surface roughness is applied through the Geometry Node Modifier by combining noise textures at different scales to obtain the desired result. This step is preferable to be done after the craters are applied on the mesh, in order to obtain a homogeneous surface with a more realistic blending of the craters.

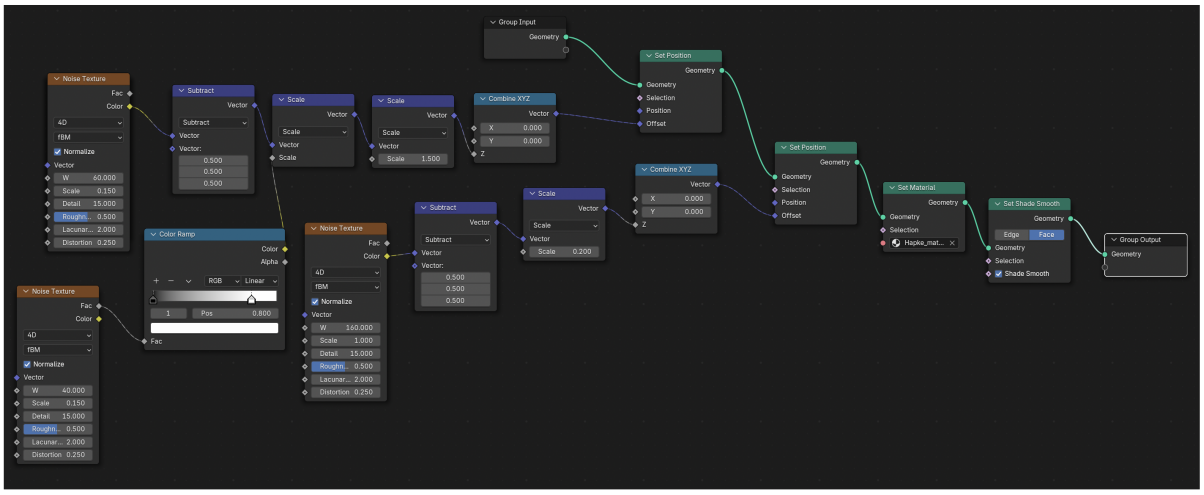


Figure 1: Geometry Nodes Small Body Surface

2.1.2 Craters

The second step consists of applying artificial craters on the smooth high-resolution mesh. Two methods can be adopted for the generation of synthetic craters.
The first method is presented in the work of M.Pugliatti and M.Maestrini [19] and it is based on patching craters on the surface as different objects by using height maps of real existing Earth craters [1]as texture maps, and the Shrinkwrap Modifier. An initial calibration is required in Paint Mode for a correct blending; after that, the same crater can be used multiple times by simply applying shape deformations on the three axis. This method can achieve a high degree of realism, especially for large craters, but becomes quite time and resource consuming when many craters need to be added to the surface.
For these reasons, a second method has been implemented, taking advantage of Blender's Python integration. Instead of adding external objects, this approach directly sculpts the base mesh towards the desired crater shape through a Python script, fully automating the process.
This strategy provides several advantages:

- **Efficiency**: a large number of craters with varying size, depth, ellipticity, and rim shape can be generated in seconds without manual intervention.
- **Lightweight geometry**: since no new meshes are added, the polygon count of the model remains low as the weight of the model
- **Scalability**: countless tiny craters can be applied across the surface, better reproducing the crater saturation typical of small bodies, something impractical with the first method.
- **Flexibility**: crater parameters (e.g., radius, depth, rim height, ellipticity, noise perturbation) can be randomized or controlled, enabling procedural diversity and more realistic crater distributions.

In this work, both methods are employed: the first for very large craters, where detailed shape realism is crucial, and the second more extensively for smaller craters, where high density and variability are required.
The faces of the mesh affected by the transformation are marked and grouped together in a Vertex Group; this allows them to be recognized as a different feature with respect to the surface, as will be explained in the next subsection.
Vertex Group is a powerful tool and can be exploited to group and mark selected areas on the mesh. It can be exploited to segment craters on acquired refined meshes of real space bodies, a halfway approach between

[1]https://tangrams.github.io/heightmapper/

automatic labeling and by-hand labeling. Indeed, once the mesh is segmented, countless masks can be retrieved. The mesh of Vesta is segmented by using this approach as shown in Figure 2, in order to include more realistic shapes in the dataset; a brush is used to manually assign weights to the craters on the mesh and group them in the same Vertex Group.

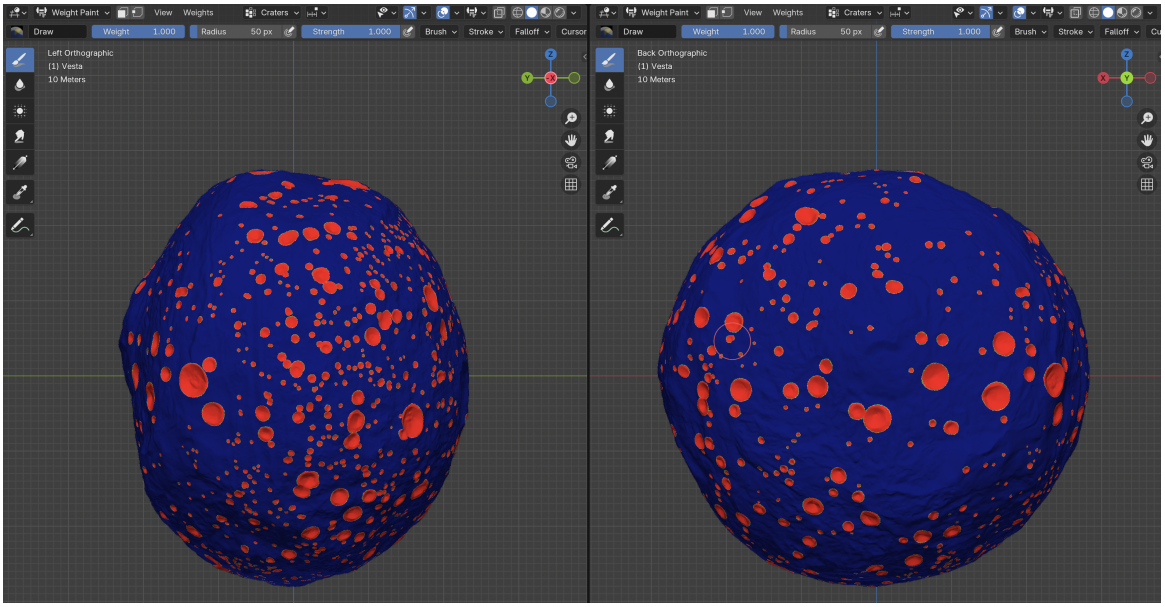


Figure 2: Vesta's mesh segmentation in Weight Paint

2.1.3 Boulders

The third step consists of placing boulders on the surface: the Rock Generator Add-on is exploited for generating a random set of rocks, variable in size and shape. Two categories are set: big boulders and small boulders. For the very tiny boulders, icospheres are used to obtain a more realistic light impact effect from long distances. The Particle System is used for spreading boulders on the surface and random variation in size and rotation is set. Moreover, Paint Mode can be exploited for a non-uniform placing of the boulders, by assigning paint weights to the surface mesh; in this way, for instance, the user can avoid placing boulders in the craters or you can create more boulders packed zones.

2.2. Automatic labeling

Automatic labeling is the key point in adopting this pipeline in the generation of synthetic images, as it allows the generation of the image–mask pairs to produce the labeled dataset. The morphological features chosen for the segmentation task are:

- Background/Shadows (Class: 0)
- Surface (Class: 1)
- Craters (Class: 2)
- Boulders (Class: 3)

In order to be able to identify each feature during the rendering, an identifier index is assigned to each feature. The identifier index chosen is the Material Index, as it indirectly allows to identify features belonging to the same object/mesh too through the use of Vertex Groups. A different material index is assigned to each material, and a different material (or even a copy of the same material) is assigned to each object, or to each Vertex Group if the features belong to the same mesh (this is the case of craters sculpted on the surface or object merged with surface mesh).
In Blender Compositor Material Indexes are the inputs for ID Mask nodes, through which logical masks for background, surface, craters, and boulders are extracted during rendering. The masks extracted do not take into account shadow areas, so other information need to be retrieved by using Light Diffuse Indirect and Color Pass. Logical operations are performed between the nodes in order to obtain the desired masks as displayed in Figure 3.

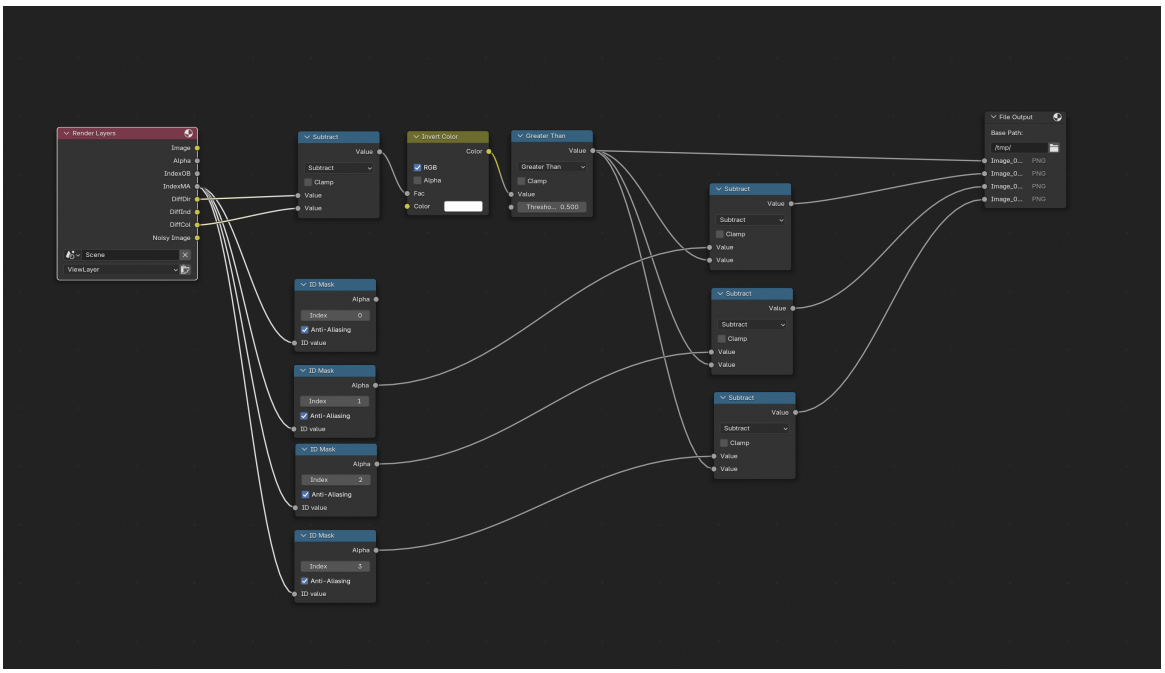


Figure 3: Compositor Nodes

2.3. Shader

The study of this section is guided by the work done by Jacopo Villa et al. [25]. The simulation of planetary surface imagery is fundamental for tasks such as optical navigation and scientific exploration. These applications demand high-fidelity synthetic images that authentically reproduce the unique properties of planetary surfaces, including reflectance, albedo, and topography. To achieve this level of accuracy, the rendering pipeline relies on Physically Based Rendering (PBR). PBR models the behavior of light and the materials interacting with it based on physical principles, capable of producing images nearly indistinguishable from real life, provided sufficient computational resources are available. The rendering engine chosen to generate high-fidelity images is Blender Cycles. Cycles is a rendering engine based on path tracing, which accurately simulates light interaction phenomena, including reflections from multiple surfaces (indirect illumination), thereby simulating realistic illumination even in complex scenes.
Path tracing is a Monte Carlo method that propagates a set of random light rays backwards from the camera, to the surface, and reflected off the surface until they reach the light source. This reverse propagation strategy is used to avoid unnecessary computation, as only rays that start at the camera can possibly affect the final image.
The light interaction computation process in Cycles proceeds as follows:

1. Multiple rays are generated per pixel from the camera focal point.
2. When a ray intersects a surface, both direct and indirect illumination effects are computed by evaluating the surface's Bidirectional Reflectance Distribution Function (BRDF).
3. Direct illumination is calculated by tracing a shadow ray directly from the intersection point toward the light source. If unblocked, direct illumination is , otherwise it means that point is in shadow.
4. Indirect illumination is evaluated by randomly sampling a bounce direction from the BRDF; the process of finding and evaluating intersection events is repeated, simulating multiple bounces.
5. The light propagation terminates when a bouncing ray either encounters the light source, leaves the scene, or reaches a pre-set maximum number of bounces.

Once all valid paths (from camera to light source) are computed, each path is re-evaluated forward (from the light source to the camera) to determine how the light radiance is affected by interactions with the scene (such as attenuation and indirect lighting effects). Path tracing, following this pipeline, provides unbiased rendering results and is fully capable of simulating global illumination (both direct and indirect lighting). The accuracy of the final rendered image increases with the number of samples per pixel as it works as a Monte Carlo method. Planetary surfaces, especially those of small bodies like asteroids, are typically covered by regolith, a layer composed primarily of fine particles, dust, and small grains. Regolith can be conceptualized as a layer of irregularly shaped, randomly oriented particles. This unique structure causes repeated scattering of light within the medium, beneath the surface [7]. Due to the subsurface scattering effects and the unique structure of the regolith, common BRDFs, such as the Lambertian model or even the Oren-Nayar model [14], are unsuitable for

accurately modeling regolith reflectance properties. For example, the Lambertian model describes reflectance simply via a cosine law and fails to capture the flattening of the surface appearance observable at low sun phase angles.

In this work, the Hapke model with 5 parameters is implemented [6] (see Appendix A.1). The Hapke model is often considered the gold standard for modeling reflectance from regolith surfaces. It acknowledges the discrete nature of the regolith particles and incorporates properties related to the particles and their structure, such as surface roughness, density, and porosity. It also explicitly accounts for multiple scattering events within the regolith layer.

The most significant impact of microscopic modeling within the Hapke model is the ability to capture the **Shadow Hiding Opposition Effect (SHOE)** or opposition surge. When regolith is observed at a high sun phase, the microscopic shadows cast by the particles make the surface appear darker. Conversely, when observed at a zero or very low sun phase, these shadows are hidden behind the lit surface, resulting in a sharp increase in surface reflectance.

To integrate the Hapke reflectance model into the Cycles engine, the software's advanced shader programming capabilities and its node-based interface are utilized.

The implementation of complex BRDFs like the Hapke model is achieved using the Open Shading Language (OSL) [1]. OSL is a programming language specifically designed for crafting shaders and defining custom BRDFs that are not natively incorporated in Blender Cycles. The OSL code defines the reflectance function, including its necessary inputs and outputs, which Cycles then uses during the path-tracing computation to determine the surface appearance. However, it must be noted that using OSL in Cycles currently disables GPU computing.

The Node Editor in Blender's Graphical User Interface (GUI) is used to define the material properties, providing a graphical representation of the data flow. The Node Editor for this study is organized in three main groups:

- Observing Geometry Information: Nodes that store the camera position and light direction which are necessary inputs for the BRDF calculation. These values can be automatically updated at runtime using Blender's drivers and they are used to compute the inputs of the Hapke model.
- Surface BRDF: This contains the OSL node programmed with the Hapke reflectance function. This node is connected to the geometry inputs and outputs.
- Surface Albedo Map: This data is fed into the BRDF node to account for local variations in surface albedo. It is usually imported in the form of a texture map, however in this case a random map is generated, since it was not intended to use any specific map. A Noise Texture addresses the generation of the map, and a minimum and a maximum albedo value are set leveraging on a mean value and a percentage of variation.

This Node Editor setup ensures that the shading and material properties defined (Hapke BRDF) are dynamically evaluated and applied to the 3D object by the rendering engine during the rendering process.

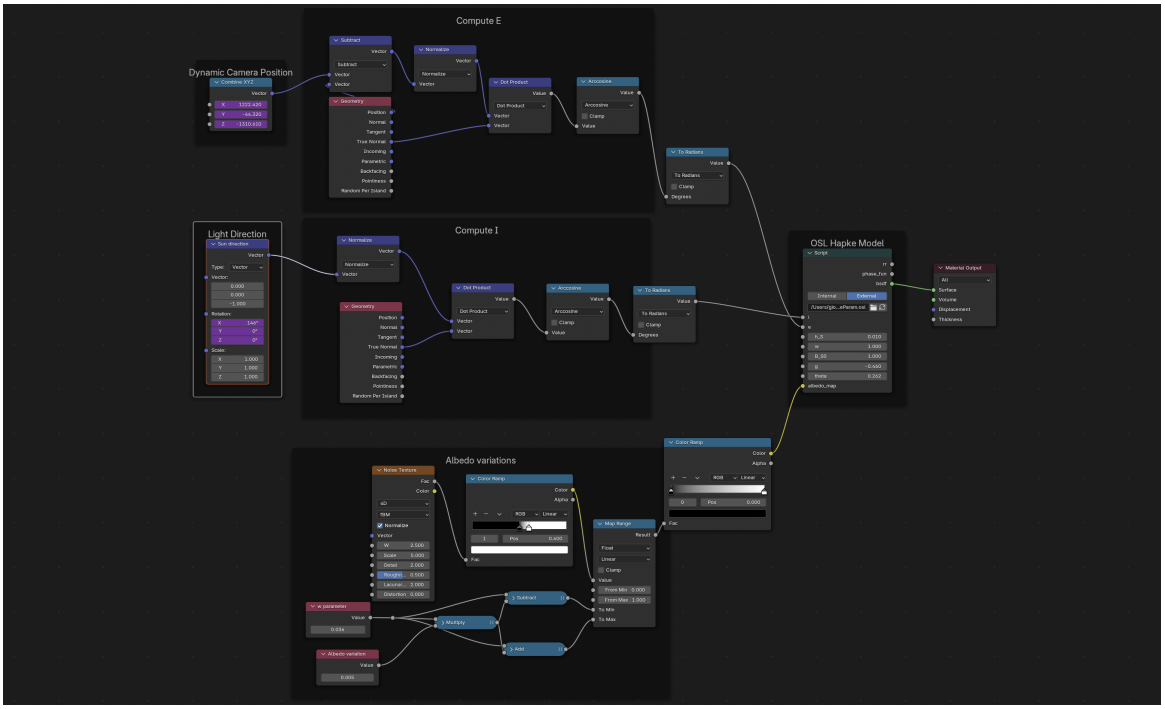


Figure 4: Shader

2.4. Rendering setup

In order to generate synthetic datasets suitable for training and testing convolutional neural networks on small body segmentation, the rendering configuration in Blender Cycles is set in the following way:

- **Image Resolution**: All rendered images are produced at a resolution of 1024×1024 pixels to match the sensor resolution of typical space optical navigation cameras, ensuring comparability between synthetic and real datasets.
- **Color Depth**: Images are saved with an 8-bit grayscale color depth to limit images weight and because it is the format of most mission imagery available in planetary science archives.
- **Number of samples**: A maximum of 256 samples per pixel is employed as trade-off between rendering realism and rendering time
- **Max Light Bounces**: This parameter is set to 1 for physical context reasons. Indeed, there is a single solar source and indirect light contribution are minimal due to the absence of atmosphere. This avoid unnecessary computation and reduce rendering time too, even if certain subtle light-scattering phenomena may be neglected.

Following the instruction of the work by M.Pugliatti and M.Maestrini [19], a cloud of random camera positions is generated around the celestial body, by sampling uniformly in a spherical shell of radius span $[0.4D_0, 1.4D_0]$ where D_0 is the approximate range at which the asteroid saturates the Field of View of the camera, which is assumed to be $10deg$. Moreover, an ideal pointing to the center of the camera is imposed.

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

The sun direction is fixed and it is always on the same side of the camera to avoid off-nominal illumination conditions, where the celestial body is completely in shadow.

2.5. Synthetic Image-Mask Pairs Generation

The presented procedural framework is employed to generate synthetic datasets for the training of the proposed neural networks (See 4).
A total of eight small bodies were selected from the 3D Asteroid Catalogue [2] as reference for dataset generation:

- **(67P) Churyumov–Gerasimenko**
- **(101955) Bennu**
- **(103P) Hartley**
- **(21) Lutetia**
- **(433) Eros**
- **(4) Vesta**
- **(41) Daphne**
- **(7) Iris**

The selection of these bodies is made in order to ensure large differentiated body shapes, diversity of surface morphology, from bilobate comets (67P, Hartley) to rubble-pile asteroids (Bennu) and crater-rich bodies (Vesta, Lutetia) , and various illumination conditions (Hapke parameters for these bodies can be retrieved from [8] [12]. The selected bodies are also representative of current and future mission targets where hazard detection and landing remain critical challenges.
The resulting synthetic dataset is automatically annotated using the proposed segmentation pipeline, producing pixel-level labels for background/shadow, surface, craters, and boulders.

2.6. Real Imagery Dataset Generation

To evaluate generalization across domain shift, a smaller set of real asteroid images is assembled. Data are collected from public mission archives [3], focusing on high-resolution optical images from:

- *OSIRIS-REx* (Bennu)
- *Dawn* (Vesta, Ceres)
- *Rosetta* (67P)

These images are manually annotated at pixel level to create ground-truth segmentation masks. The real dataset is substantially smaller than the synthetic one, collecting a total of 30 images since manual annotation is quite time consuming and to be aligned with the size of real test dataset proposed by M.Pugliatti and M.Maestrini [19].

Table 1: Summary of morphological features for the selected small bodies

Body	Large Craters	Small Craters	Boulders
(67P) Churyumov–Gerasimenko	4	/	500
(101955) Bennu	3	/	30000
(103P) Hartley	5	/	2000
(21) Lutetia	5	100	1500
(433) Eros	6	50	500
(4) Vesta	/	/	1000
(41) Daphne	5	20	1000
(7) Iris	3	50	500

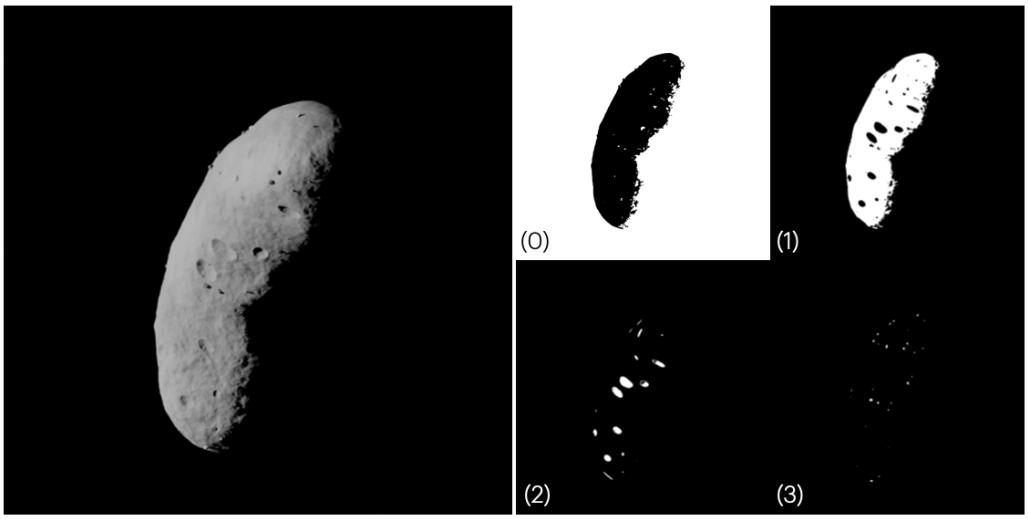


Figure 5: Original image and label masks of 433 Eros asteroid: (0) Background, (1) Surface, (2) Craters, (3) Boulders

3. Noise augmentation

One of the main challenges in training convolutional neural networks for semantic segmentation of small body imagery lies in the gap between synthetic and real mission data. Blender software does not provide the possibility to model a noise camera model, so the generated synthetic datasets are clean, with no sensor artifacts, while real images from space missions contain significant amounts of noise. This discrepancy, often referred to as domain shift, can considerably reduce the generalization capability of models trained purely on synthetic data. To mitigate this issue, two noise augmentation strategies were adopted during training: a classic artificial noise augmentation, commonly used in computer vision tasks, and a noise augmentation based on an analysis of real space imagery.

3.1. Random Noise

The first strategy involves the application of standard image perturbations available in the TensorFlow augmentation pipeline. These include:

- Random brightness adjustment with a maximum change set to 0.3
- Random contrast scaling with a contrast factor set between 0.8 and 1.2
- Gaussian noise:

$$I'(x,y) = I(x,y) + \epsilon, \qquad \epsilon \sim \mathcal{N}(\mu_n, \sigma_n^2) \tag{2}$$

  where $\mathcal{N}$ is a normal distribution with mean $\mu_n = 0.1$ and standard deviation $\sigma_n = 0.001$.
- Salt-and-pepper noise:

$$I'(x,y) = \begin{cases} 0, & \text{with probability } p/2, \\ 1, & \text{with probability } p/2, \\ I(x,y), & \text{with probability } 1-p, \end{cases} \tag{3}$$

  where $p = 0.01$.

These transformations simulate common noise conditions such as illumination variations, sensor grain, and random transmission errors. They are applied uniformly across the entire image, without distinguishing between the celestial body and the background. Although generic, this strategy effectively improves robustness against pixel-level perturbations, since the network is exposed to a wide spectrum of noise patterns and thereby enhances its generalization capability.

3.2. Real Missions Noise

The second strategy is based on a systematic analysis of noise patterns observed on real imagery past missions such as from Regolith Explorer (OSIRIS-REx) [11], Dawn [22], and Rosetta [26]. The goal is to construct an equivalent noise model that captures not only statistical properties but also structural patterns found in the data.

Images are divided into 3 different classes:

- Deep space images: nearly black frames pointing the outer space
- Fly-by images: the celestial body fully contained within the field of view
- Surface images: close-range picture of the body's terrain

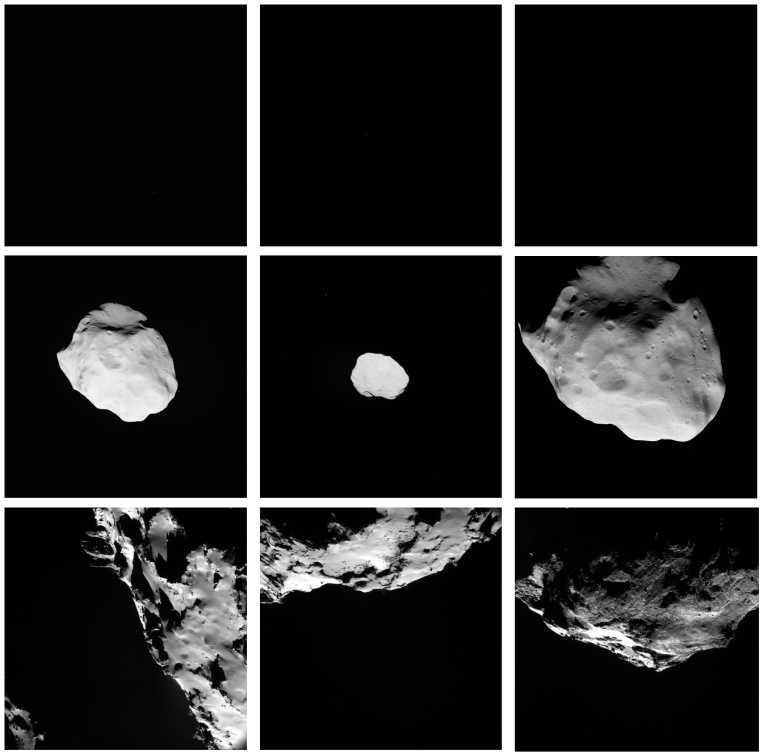


Figure 6: From the top row: deep space, fly-by, surface

For each category, noise is extracted by masking out high-intensity regions corresponding to the celestial body, followed by generating overlapping black patches. On these patches, the mean intensity and standard deviation of the noise distribution are computed.

Class	$\text{Mean}_{(over255)}$	Std	Samples
Deep space	2.51	0.19	
Fly-by	2.55	0.22	200
Surface	2.12	2.01	

Table 2: Black Patches pixels intensity stats

The Table 2 highlights how deep space and fly-by images have very similar mean intensity values with very low standard deviation. On the other hand surface images, despite having comparable mean, have stronger local variation in noise which could be due to light-surface interactions.

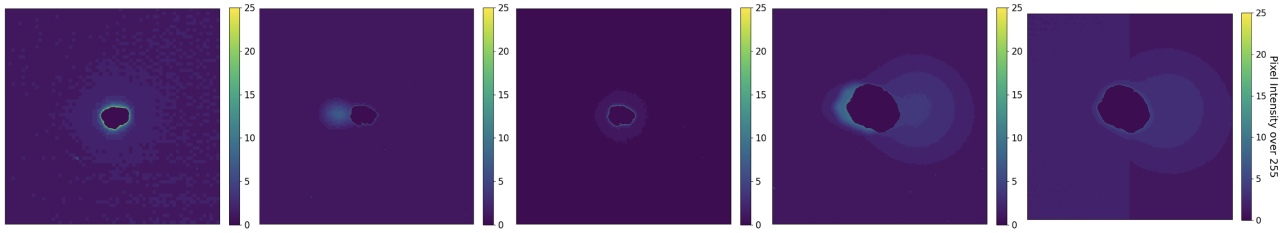
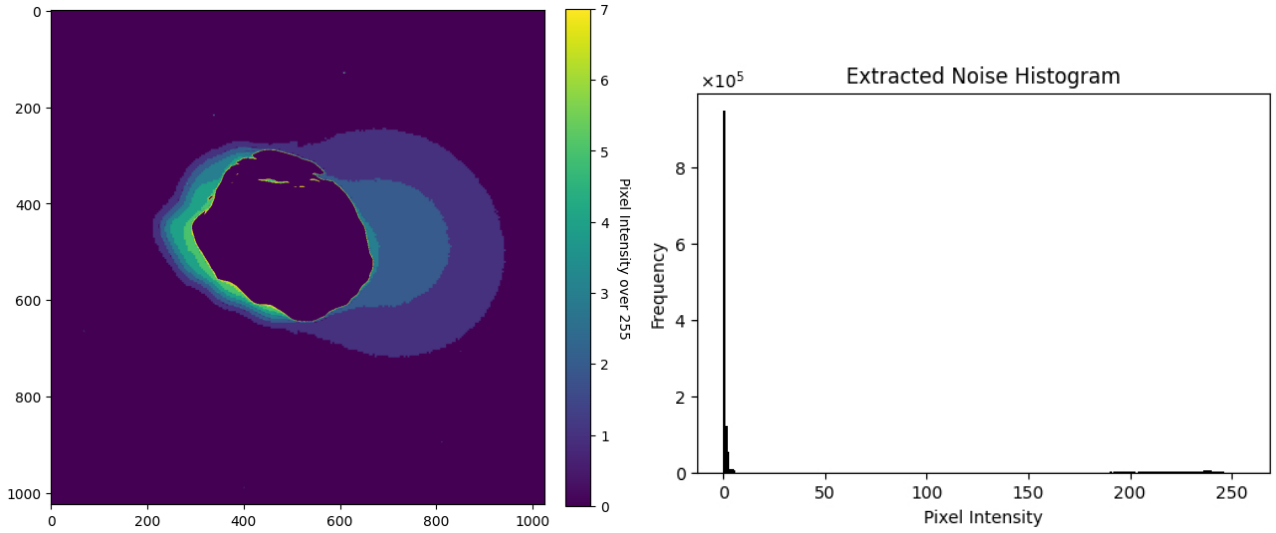


Figure 7: Fly-by mask examples

Moreover, by inspecting the extracted fly-by masks, recurrent particular shapes of the noise can be recognized:

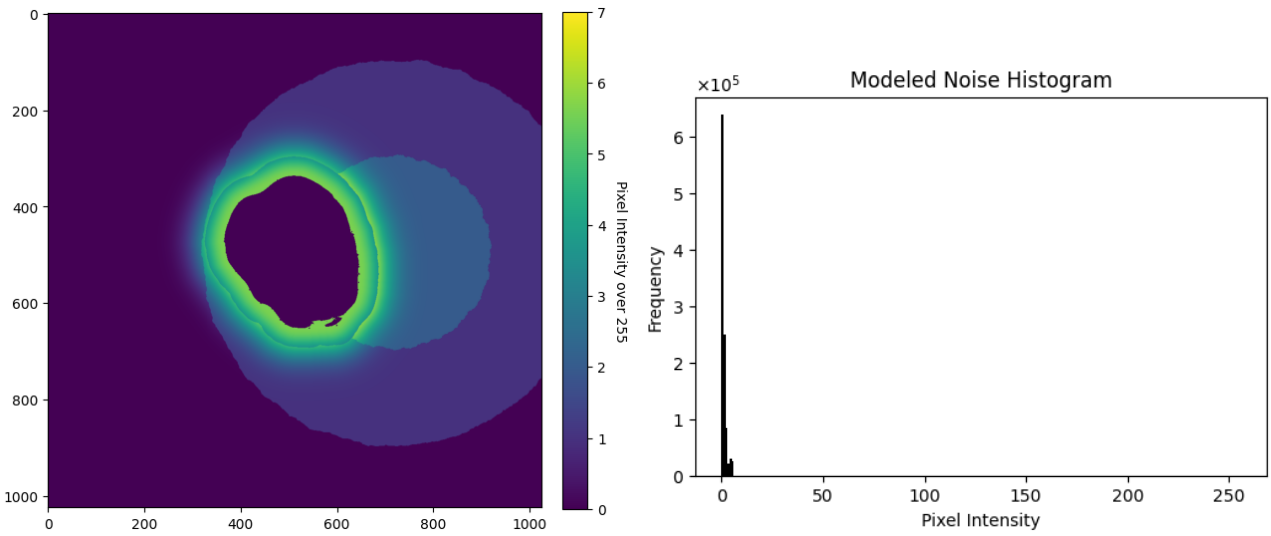
- halo-shaped noise around the lit body with high intensity
- trail noise

The halo shape around the body suggests light scattering in the camera sensor, due to the high contrast of the lit surface against the background, while the trail noise resembles a motion blur effect due to spacecraft motion around the body.
Based on these observations, a noise generator was developed to replicate both the statistical distribution and the structural characteristics of the observed noise. This generator was applied on synthetic images during training, with the aim of bridging the gap between synthetic and real mission data, thereby enhancing the robustness of the trained networks. Figure 8 and Figure 9 show a comparison between the noise extracted from a real image and the generated noise applied on a synthetic image.



(a) (b)

Figure 8: (a) Extracted Noise Intensity, (b) Extracted Noise Histogram



(a) (b)

Figure 9: (a) Modeled Noise Intensity, (b) Modeled Noise Histogram

4. Artificial Neural Networks for Segmentation

The design of the network architecture is a central element in semantic segmentation, as it directly influences both the representational power of the model and its ability to generalize to unseen data. In this work, the objective is to achieve robust segmentation of small body surfaces by training exclusively on synthetic datasets, while ensuring good performance on real mission imagery. The constraints of computational resources and the need for generalization under domain shift require a careful balance between model complexity and efficiency. For this reason, two architectures are selected for this study: a reduced-size U-Net, used as the baseline, and a reduced-size Multi-Residual U-Net, employed to test the benefits of residual and multi-scale feature learning.

4.1. U-Net

U-Net has become a standard architecture for semantic segmentation in scientific imaging tasks due to its encoder–decoder structure with skip connections [21]. The encoder captures progressively higher-level features, while the decoder reconstructs spatial detail by combining them with low-level information from earlier layers. This architecture is particularly well suited to segmentation problems with limited annotated data, as it achieves strong performance even when trained on relatively small datasets. In this study, a reduced-size U-Net was chosen as the baseline for several reasons:

- Computational efficiency: full U-Net models are resource-intensive, both in terms of memory and training time. Reducing the number of layers and parameters allows training on larger synthetic datasets and accelerates experimentation without sacrificing the core representational structure of the network.
- Limited dataset size: given the limited availability of real asteroid imagery, a smaller U-Net reduces the risk of overfitting and may generalize better under domain shift when compared to deeper models with excessive capacity.
- Baseline interpretability: U-Net is well understood in the literature, with extensive benchmarking across domains. Using it as a baseline ensures comparability with prior work in planetary image segmentation.

Despite these advantages, the reduced U-Net has limitations, as its relatively shallow structure may hinder its ability to capture complex morphological features such as small craters or boulders, which often exhibit subtle local patterns. Moreover, the standard U-Net simple skip connections may not be sufficient to propagate robust features across scales in highly noisy or heterogeneous images.

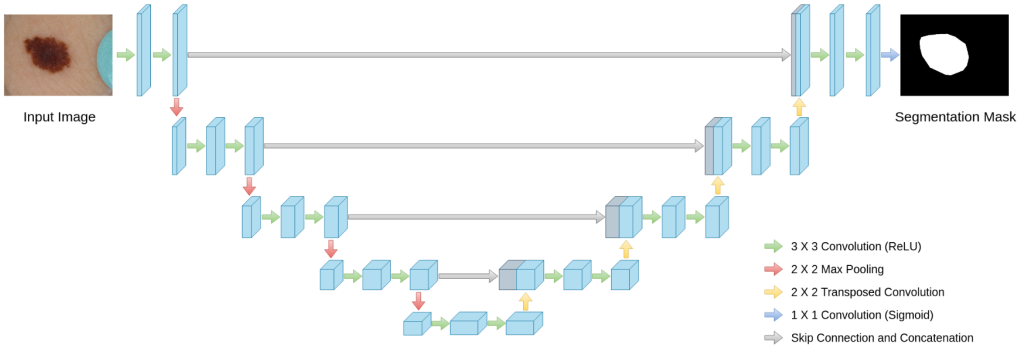


Figure 10: Classic U-Net architecture [9]

4.2. Multi-Residual U-Net

To address these limitations, Multi-Residual U-Net is introduced as a second architecture [9]. This network extends the U-Net design by incorporating residual connections and multi-scale convolutional blocks. The main advantages of this new network are:

- Residual learning: residual paths are designed to cover the semantic gap between encoder and decoder connected layers, therefore, in order to mitigate the the disparity between the encoder-decoder features, convolutional layers along the shortcut connections are implemented (see Figure 11).

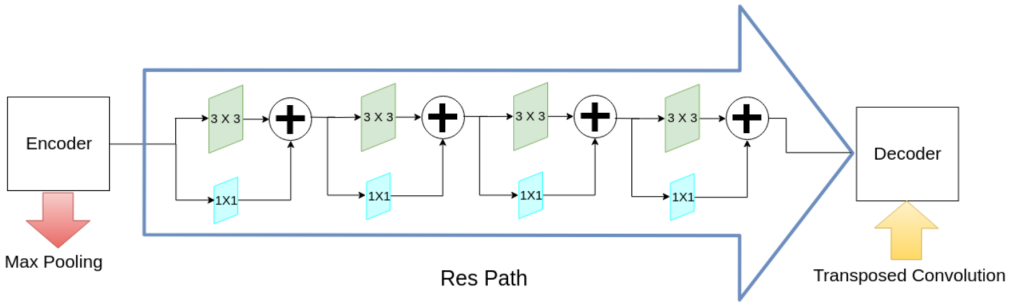


Figure 11: Res path [9]

- Multi-scale feature extraction: by using convolutions of different kernel sizes within each block, the Multi-Residual U-Net captures spatial patterns at multiple resolutions simultaneously (see Figure 12. This is advantageous for small body segmentation, where features of interest range from large shadowed regions to small-scale boulders.

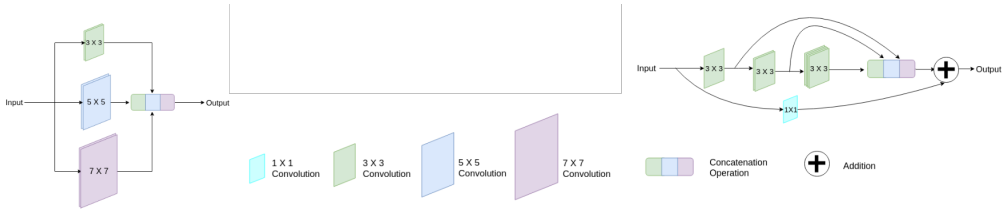


Figure 12: On the left the concept behind the MultiRes block; on the right the actual implementation that involves the concatenation of increased kernel sizes convolutions for memory saving [9]

For these reasons, the reduced-size Multi-Residual U-Net is chosen as an extension of the baseline. Reducing its depth and parameter count ensures computational feasibility and fair comparison with the baseline U-Net, while preserving the architectural innovations that may yield performance improvements.

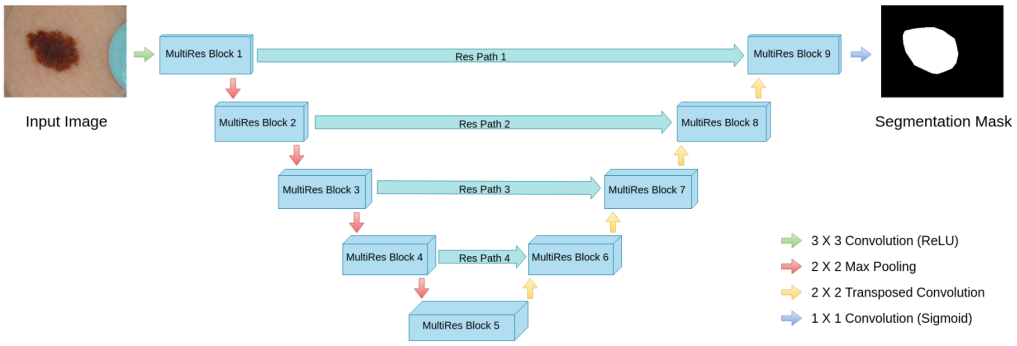


Figure 13: Classic MultiRes U-Net architecture [9]

4.3. Training Architectures

In order to ensure a fair comparison between the reduced U-Net and the reduced Multi-Residual U-Net, both architectures are scaled down in size. Specifically, the number of downsampling operations and corresponding feature maps are adjusted such that the spatial dimensions at the bottleneck layer (height and width) are identical in both models. This design choice guarantees that the two networks operate on the same level of feature abstraction at their deepest layer, thereby allowing performance differences to be attributed to the architectural design (standard skip connections and convolutional blocks vs Res paths and MultiRes blocks).
The classical networks architectures are modified to provide:

- **Input shape**: $(1024, 1024, 1)$
- **Bottleneck shape**: $(64, 64, depth)$
- **Output shape**: $(1024, 1024, 4)$ with Softmax Activation

	Trainable params	**Non-Trainable params**	**Total params**
U-Net	1940868	0	1940868
MultiRes U-Net	7244684	24520	7269204

Table 3: Networks number of parameters

4.4. Training Setup

The choice of loss function, performance metrics, and optimization strategy plays a critical role in the effectiveness of semantic segmentation models. These elements must be chosen to mitigate class imbalance, ensure stable training, and provide meaningful evaluation of model performance. In this work, weighted categorical cross-entropy is employed as the loss function, with accuracy, and Mean Intersection over Union (MIoU) and per class Intersection over Union (IoU) as evaluation metrics. Different optimization schemes are applied for the baseline U-Net and the Multi-Residual U-Net to account for their structural differences and training dynamics.

4.4.1 Loss Function

A key challenge in asteroid surface segmentation is class imbalance: background and surface pixels dominate the images, while craters and boulders are underrepresented. This imbalance can bias the model toward predicting the majority classes and neglecting minority features, which are often the most relevant for hazard detection. To mitigate this, the categorical cross-entropy loss is weighted according to the inverse frequency of each class. The class dictionary weights are computed from the training dataset and applied as multipliers in the loss function, ensuring higher penalization for misclassified crater and boulder pixels.
Class weights are computed for each class c [17] according to:

$$w_c = \frac{N}{C \cdot n_c} \tag{4}$$

where N is the total number of pixels in the dataset, C is the number of classes, and n_c is the number of pixels belonging to class c. This assigns higher weights to minority classes, forcing the network to penalize misclassifications on rare but important features (e.g., craters, boulders). This strategy improves the model's sensitivity to small or rare features while preserving stability for abundant classes.
After computing the weights, it is observed that the values for *craters* and *boulders* are very similar. However, *craters* are inherently more difficult to detect due to their variable size, faint boundaries, and similarity to surface textures. For this reason, the weights of the two classes are intentionally swapped, assigning a higher weight to craters; this decision is made to improve the model sensitivity to craters, while maintaining adequate attention to boulders.

Table 4: Class dictionary weights used in the weighted categorical cross-entropy.

Class	**Weight**
Background / Shadow	0.48
Surface	0.56
Craters	15.81
Boulders	12.10

The weighted categorical cross-entropy is defined as:

$$L = -\sum_{c=1}^{C} w_c \, y_c \, \log(\hat{y}_c) \tag{5}$$

where C is the number of classes, y_c and $\hat{y}_c$ are the ground-truth and predicted probabilities for class c, and w_c is the weight assigned to class c.

4.4.2 Performance metrics

To evaluate segmentation performance, three metrics are adopted:

- **Accuracy**: measures the overall proportion of correctly classified pixels.
  Pixel accuracy measures the proportion of correctly classified pixels over the total number of pixels [13]:

$$\text{Accuracy} = \frac{\sum_{c=1}^{C} TP_c + TN_c}{\sum_{c=1}^{C} (TP_c + TN_c + FP_c + FN_c)} \tag{6}$$

  where TP_c, FP_c, and FN_c denote the true positives, false positives, and false negatives for class c, respectively. While simple and intuitive, accuracy can be misleading in imbalanced datasets, since performance on dominant classes can overshadow errors on minority ones.
- **IoU**: computed per class as the ratio of the intersection and union of predicted and ground-truth masks [13]. IoU provides a more informative measure of class-specific segmentation quality. The per class IoU_c is defined as:

$$\text{IoU}_c = \frac{TP_c}{TP_c + FP_c + FN_c} \tag{7}$$

  where TP_c, FP_c, and FN_c are the true positives, false positives, and false negatives for class c, respectively.
- **MIoU**: the average IoU across all classes, balancing the evaluation between majority and minority classes. This is considered the most reliable metric in this study, as it reflects the ability of the model to segment both abundant and rare morphological features. The MIoU is computed as the average IoU over all C classes:

$$\text{MIoU} = \frac{1}{C} \sum_{c=1}^{C} IoU_c \tag{8}$$

4.5. Optimization and Training strategy

The training strategy is adapted to each network architecture to balance convergence speed and learning stability. A summary is provided in Table 5.

Table 5: Training configurations for U-Net and Multi-Residual U-Net.

Model	Optimizer	LR	WD	Early Stopping	LR Scheduler	Checkpoint Criteria
Reduced U-Net	Adam	1×10^{-3}	–	Yes	ReduceLROnPlateau	Best val.loss & mIoU
Multi-Residual U-Net	AdamW	1×10^{-4}	1×10^{-4}	Yes	ReduceLROnPlateau	Best val.loss & mIoU

Both networks employ early stopping to prevent overfitting, and ReduceLROnPlateau to lower the learning rate when validation loss stops decreasing. At each epoch, two models are checkpointed: one corresponding to the lowest validation loss and one to the highest validation MIoU.
Early stopping is a regularization technique used to prevent overfitting. During training, the model typically continues improving its performance on the training set while its performance on the validation set may plateau or even degrade after a certain number of epochs. Early stopping monitors a validation metric (in this case validation loss and MIoU), and stops training if no improvement is observed after a predefined number of epochs (patience hyperparameter), since prolonged training could lead to memorization of training set patterns obstructing the generalization and so the learning itself.
The Learning Rate (LR) is one of the most sensitive hyperparameters in deep learning. A fixed learning rate can lead to suboptimal convergence:

- If the LR is too high, the optimizer oscillates and fails to converge
- If the LR is too low, training becomes excessively slow and may get stuck in poor local minima.

The ReduceLROnPlateau scheduler [20] dynamically reduces the learning rate when a monitored metric (in this case validation loss and MIoU) stops improving. This allows the model to use a higher learning rate in the early epochs for faster convergence, and progressively smaller learning rates in later stages for fine-tuning.
Weight decay is a form of L2 regularization applied during optimization [5]. Its purpose is to discourage the network from learning excessively large weights, which can lead to overfitting and poor generalization, especially when training on synthetic data. Mathematically, the loss function with weight decay is:

$$L' = L + \lambda \sum_{i=1}^{P} \omega_i^2, \tag{9}$$

where L is the original task loss (e.g., weighted categorical cross-entropy), λ is the weight decay coefficient, ω_i are the trainable parameters, and P is the total number of parameters in the model. This additional term penalizes large weights and encourages smoother and more stable solutions.
The weight decay coefficient directly controls the strength of the regularization term. If it is set too high, the optimizer overly penalizes large weights, leading to underfitting and preventing the network from learning meaningful representations. On the contrary, if it is too low, the regularization effect becomes negligible and the model may overfit to the training data.
All models are trained from scratch, without the use of pretrained weights or specialized initialization schemes. This choice directly influenced the selection of both the learning rate and the regularization strategy:

- **U-Net**: A higher learning rate of 10^{-3} is selected together with the Adam optimizer and no weight decay. The lower number of parameters already reduces the risk of overfitting and allows the network to benefit from more aggressive parameter updates. Since training started from random initialization, a relatively large learning rate ensures faster convergence to meaningful representations. Weight decay is not applied, as over-regularization could obstruct the learning capacity of the model.
- **MultiRes U-Net**: A lower learning rate of 10^{-4} is adopted, combined with the AdamW optimizer and weight decay $\lambda = 10^{-4}$. The deeper and larger architecture, initialized from scratch, requires a smaller learning rate to prevent unstable updates in the early stages of training. The higher number of parameters also makes the model more prone to memorizing training patterns, thus weight decay was introduced as an additional regularizer. There is no written rule on how to choose the correct value so, after preliminary experiments, the value of 10^{-4} is chosen.

This training framework is designed to account for training from scratch, severe class imbalance and the need for robustness under domain shift between synthetic and real imagery. Weighted categorical cross-entropy directly

addresses imbalance, while the inclusion of IoU-based metrics ensures that performance on minority classes is not overlooked. The use of different optimizers and hyperparameters for U-Net and Multi-Residual U-Net reflects a tailored approach, recognizing that deeper and more complex architectures require stricter regularization and slower learning rates to achieve stable training.

5. Results

The Small Body image-mask pairs rendered in Section 2.5 are employed in the generation of three synthetic training datasets.
The first dataset **D1** includes a total of six bodies, all of them used either in the training set and in the validation set. **Random Noise** augmentation (see Section 3.1) is applied on the 20% of the entire dataset. The second dataset **D2** is equal to dataset **D1** but **Real Missions Noise** augmentation is used instead. The third dataset **D3** includes a total of eight bodies, six of which compose the training set, while the last two compose the validation set. The larger amount of images and the presence of unseen bodies in the validation set should improve the training quality by preventing overfitting and stressing the network toward better features generalization. Moreover, both types of noise augmentation are applied on dataset **D3**, in equal distribution with percentages of 33%. A summary of the characteristics of the three training datasets is shown in Table 6.

Dataset	Training Bodies	Number of Images	Validation Bodies	Number of Images	Noise Augmentation
D1	(Eros,Vesta, Lutetia,67P, 103P,Bennu)	8392	(Eros,Vesta, Lutetia,67P, 103P,Bennu)	2394	**Random Noise** (20%)
D2	(Eros,Vesta, Lutetia,67P, 103P,Bennu)	8392	(Eros,Vesta, Lutetia,67P, 103P,Bennu)	2394	**Real Missions Noise** (20%)
D3	(Eros,Vesta, Lutetia,67P, 103P,Bennu)	11188	Iris,Daphne	3198	**Random Noise** (33%) + **Real Missions Noise** (33%)

Table 6: Datasets details summary

Three test cases are proposed in order to fulfill the objectives of this study:

- **Test Case 1**: Training of two baseline U-Net models respectively on datasets **D1** and **D2** for investigating how different types of augmentation affect the segmentation task.
- **Test Case 2**: Training of baseline U-Net model on dataset **D3** to evaluate the benefit in segmentation performance when using both type of noise augmentation, and to assess improvements in the robustness of the network when trained with unseen validation bodies.
- **Test Case 3**: Training of MultiRes U-Net on dataset **D3** to assess whether more complex architectures are more robust and capable to better generalize under domain shift conditions with respect to the baseline U-Net.

All the models are evaluated on the manually labeled Real Imagery Dataset (see Section 2.6 for details) and compared with metrics of Accuracy, MIoU and per-class IoU defined in Section 4.4.2. At the end the results are compared with the outcome of the work done by M.Pugliatti and M.Maestrini [19] to assess potential improvements.

5.1. Test Case 1

In the first test case, two U-Net models are evaluated, one trained with **Random Noise** augmentation (Figure 14), and one trained with **Real Missions Noise** augmentation (Figure 15).
In both cases, as expected from the class imbalance issue (see Section 4.4.1), background and surface achieve the best segmentation results. Despite the mitigation provided by applying balancing class weights (reported in Table 4), the most represented classes in the dataset are still the best segmented.
Craters and boulders are trickier to classify, not only because they are much less represented, but also because they have a more complex semantic meaning, making them more sensitive to domain shift.
Moving on the comparison between the two models by looking at Table 7, it immediately shows up the impact of the different types of noise augmentation on the performance of the segmentation task. Despite the two

Network	Dataset	Accuracy	MIoU	IoU Background	IoU Surface	IoU Craters	IoU Boulders
U-Net	**D1**	**0.93**	**0.48**	0.97	**0.68**	**0.16**	0.10
U-Net	**D2**	0.92	**0.48**	**0.98**	0.64	0.15	**0.15**

Table 7: Summary of accuracy, MIoU and per class IoU for each model

networks have the same MIoU, the network trained on dataset **D1** has higher value of IoU on class 0 and class 3. Complementary to the first model, the model trained on dataset **D2** shows better performance on class 1 and class 2. These results highlight how the **Random Noise** augmentation is effective for the segmentation of surface and craters classes. The noise analysis showed that the noise on the surface patches has a higher standard deviation than the noise present in deep space or fly by patches; probably the stronger intensity of the random noise makes the model more robust in surface pixel classification and allows the model to generalize better on the crater class too. **Real Missions Noise** augmentation, on the contrary, improves the segmentation of the background/shadowed areas; this is due to the characteristics of the noise extracted from real images, indeed it derives from the analysis of dark region patches and it is a meaningful local distribution.

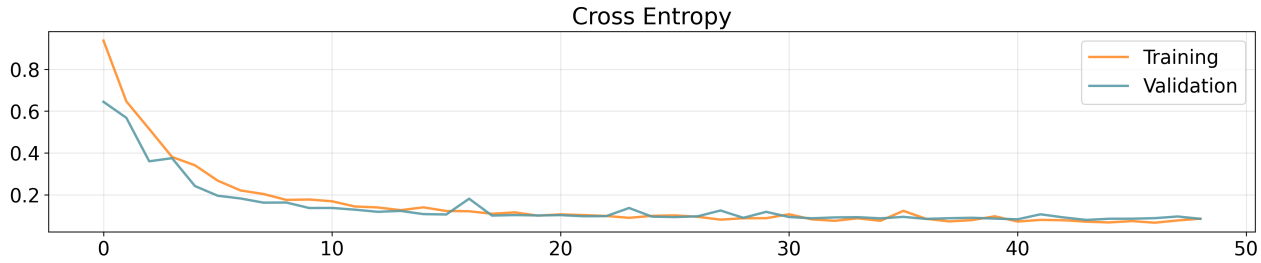


Figure 14: U-Net model trained on **D1**: training losses over epochs

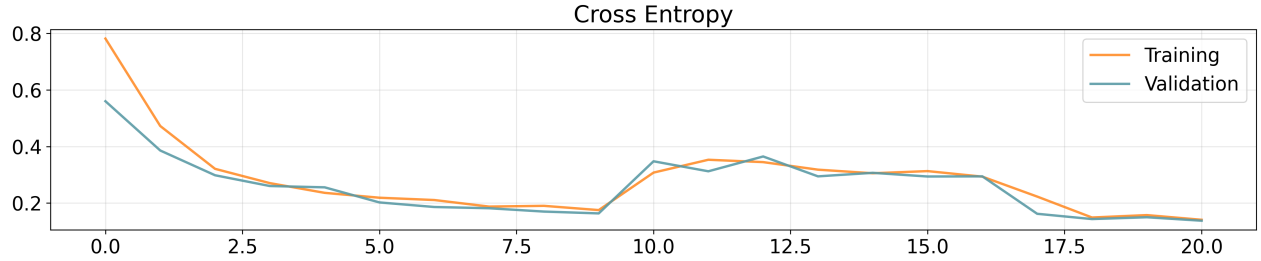


Figure 15: U-Net model trained on **D2**: training losses over epochs

5.2. Test Case 2

Network	Dataset	Accuracy	MIoU	IoU Background	IoU Surface	IoU Craters	IoU Boulders
U-Net	**D1**	0.93	**0.48**	0.97	0.68	**0.16**	0.10
U-Net	**D2**	0.92	**0.48**	**0.98**	0.64	0.15	**0.15**
U-Net	**D3**	**0.94**	0.46	0.97	**0.74**	0.04	0.08

Table 8: Summary of accuracy, MIoU and per class IoU for each model

In the second test case, a U-Net model is trained on **D3** dataset, where both types of noise augmentation are employed, moreover the validation set contains only bodies that are not included in the training set. The new training split should improve the robustness of the network bringing to better general performance. It is expected a blend of the segmentation performances of the first two models, by exploiting the advantages of **Random Noise** augmentation on segmentation of surface and boulders, and **Real Missions Noise** augmentation on background and boulders, resulting in an overall better inference on all the four classes. The background IoU is aligned with the first two models and a growth is registered on class 1. However, the data in Table 5.2 show poor results in segmenting craters and boulders classes. Training and validation losses can be inspected for further analysis (Figure 16).

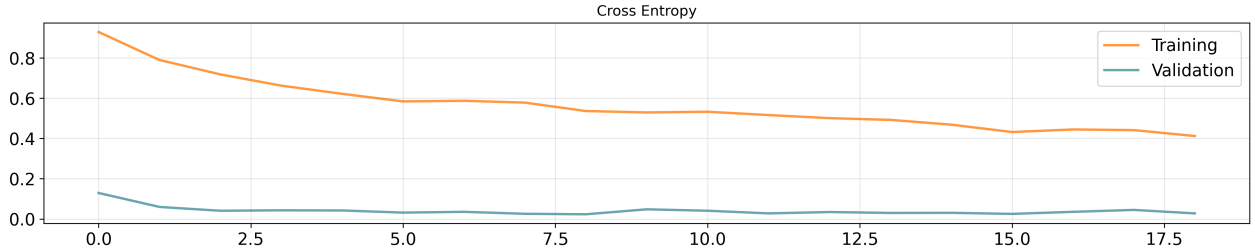


Figure 16: U-Net model trained on **D3**: training losses over epochs

Training and validation loss do not have the classical behavior that can be seen in the first test case in Figures 14 and 15, where the two losses decrease together by intersecting in many points. In this case, the validation loss is always lower than the training loss and it is near zero since the start of the training. This can happen if the validation dataset is an 'easy test', meaning that the network is not capable of learning new features because it is tested on a set that is not representative of the feature it is supposed to learn. The number of epochs shows the same too, since the early stopping stopped the learning after 16 epochs, suggesting that the learning stopped. Another reason that could explain the behavior of the two losses is the too high learning rate, that could cause too much fluctuation in the training. However, this hypothesis is discarded since the model is trained with the same learning rate of the first test case 5.1. Therefore, training and validation losses show the evidence of the presence of an imbalance in the training dataset due to a wrong splitting strategy between training and validation set. There could be an explanation about why the wrong splitting of the dataset has penalized more the class 2 and 3. The reason is that the small body models used in the validation set (Iris and Daphne) are poor of morphological features with respect to the bodies included in the training set; in particular it can be noticed that the bodies that are most representative of the bad segmented features are kept out from the validation set: Bennu for the boulders and Vesta for the craters.
In conclusion, it can be stated that the employment of dataset **D3** as training set, has increased the robustness of the model on background and surface segmentation thanks to the larger amount of images, the new training/validation splitting strategy and the inclusion of both types of augmentation, but it weakened the model as a result of the wrong selection of bodies for the splitting strategy.

5.3. Test Case 3

Network	Dataset	Accuracy	MIoU	IoU Background	IoU Surface	IoU Craters	IoU Boulders
U-Net	**D3**	0.94	0.46	**0.97**	0.74	0.04	0.08
MultiRes U-Net	**D3**	**0.98**	**0.48**	**0.97**	**0.76**	**0.12**	**0.1**

Table 9: Summary of accuracy, MIoU and per class IoU for each model

In the third test case, an advanced U-Net based architecture is evaluated after being trained on **D3** dataset. MultiRes U-Net is expected to have larger generalization capability and higher segmentation performance with respect to the baseline U-Net, as it has more parameters and above all it makes use of Res paths and MultiRes blocks, instead of simple skip connections and convolutional blocks, that should capture more meaningful latent features during the training. However, as shown in Figure 17, the same issues of the previous case (5.2) are encountered, since they are both trained on the same dataset **D3**, thus the same considerations can be assessed.

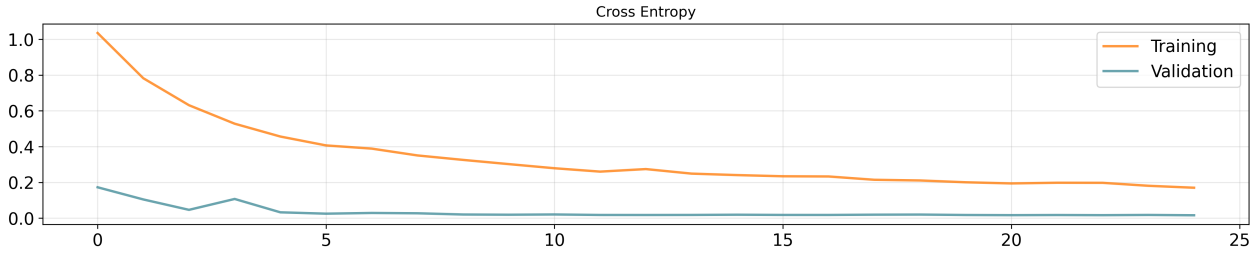


Figure 17: MultiRes U-Net trained on **D3**: training losses

The results in Table 9 show how Multires U-Net is able to achieve better segmentation performance on all the four classes despite the imbalance and bad conditioning of the dataset **D3**; background IoU is aligned with the previous models, surface IoU is further improved, and crater and boulders IoU move close to the best performance achieved in Test Case 1 (5.1).
In Table 10 the networks performances are summarized and compared in accuracy, Mean Intersection over Union ans per-class Intersection over Union.

Network	**Dataset**	**Accuracy**	**MIoU**	**IoU Background**	**IoU Surface**	**IoU Craters**	**IoU Boulders**
U-Net	**D1**	0.93	0.48	0.97	0.68	**0.16**	0.10
U-Net	**D2**	0.92	0.48	**0.98**	0.64	0.15	**0.15**
U-Net	**D3**	0.94	0.46	0.97	0.74	0.04	0.08
MultiRes U-Net	**D3**	**0.98**	**0.48**	0.97	0.76	0.12	0.1
Pugliatti & Maestrini	/	/	0.39	0.70	**0.82**	0.13	0.25

Table 10: Summary of accuracy, MIoU and per class IoU for each model

The results obtained through this study are aligned with the outcome of the work of M.Pugliatti and M.Maestrini [19] for what concerns the evaluation of synthetic trained models on real data. In order to do a comparison between the two works, some differences needs to be highlighted:

- In Pugliatti and Maestrini's work, different synthetic and real images, with a resolution of 256×256 pixels, are employed.
- In Pugliatti and Maestrini' work, five segmentation classes are chosen instead of four (background, surface, craters, boulders and terminator region).
- In Pugliatti and Maestrini' work, the networks are trained with pretrained weights picked from a small body classification network.

From Table 10 a strong increase in the IoU of background class can be noticed; the background IoU is increased from the 70% of Pugliatti and Maestrini's outcomes to a maximum of 98% in this study, with a consequent enhancement of the MIoU from 39% to 48%. Further margins of improvements are available taking into account that the models trained in this work are trained from scratch with any type of weight initialization. A proper initialization of the weight can indeed enhance the generalization by providing to the network a better latent representation since the start of training, avoiding to reach local minima and stabilizing the gradient propagation. The growth of background IoU is probably due mostly to a better choice of the classification classes rather than an effective noise augmentation strategy, indeed all the networks trained in this work perform very well in this segmentation task. Their work [19] had 5 segmentation classes instead of the four proposed by this study, and one of them (terminal region) was employed only to segment correctly the lit surface and separate it from the background. The presence of this class, in addition to make the mask generation process more complex, hinder the segmentation performance of the network. This reveal how the choice of the right classes is fundamental and how can affect the performance of the segmentation task. In this context, as ultimate analysis, a qualitative consideration about the way the network learns how to segment crater and boulder can be done. Figure 18 shows clearly how tiny craters are often classified as boulders. This happens because boulders show high contrast with respect to the surface and they are often followed by a shadow tail and at the same way little craters show

these identical properties. There is though a difference between the shadow of craters and boulders, it is the opposite concavity of the objects. In order to help the network to cover this knowledge gap, two strategies can be suggested. A first attempt can be done through the generation of crater masks that include the shadow, trying to enforce the circular shape constraint; a second attempt consists in informing the network about the light direction so that it can distinguish the direction of the surface concavity.
As a final statement, it must be pointed out that the metrics obtained by these tests are not reliable enough to assess the absolute performance of the models for many reasons. Indeed, the ground truth suffers from human error itself, the masks are labeled only by the author, who can be biased, and for the last, the set is not so large.

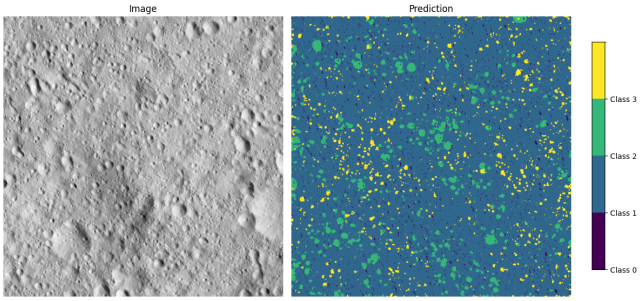


Figure 18: Example of inference on tiny craters rich terrain

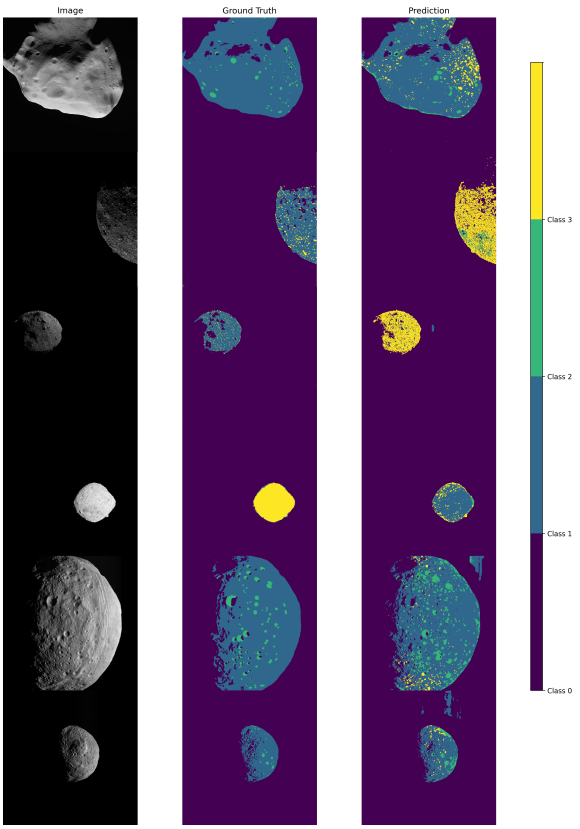


Figure 19: U-Net **D1** mosaic

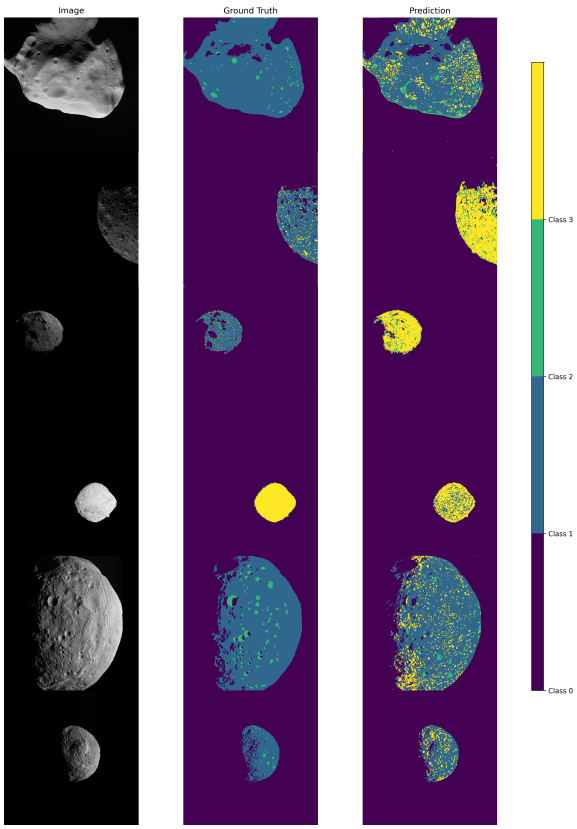


Figure 20: U-Net **D2** mosaic

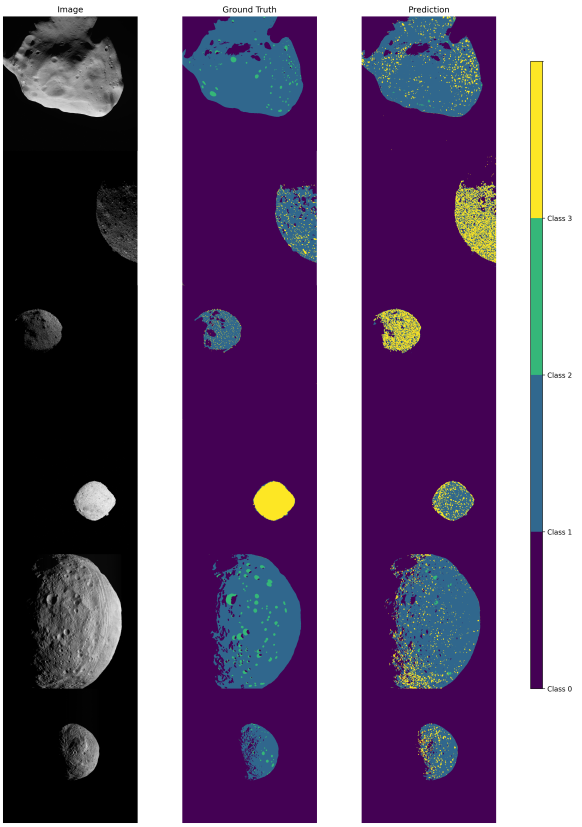


Figure 21: U-Net **D3** mosaic

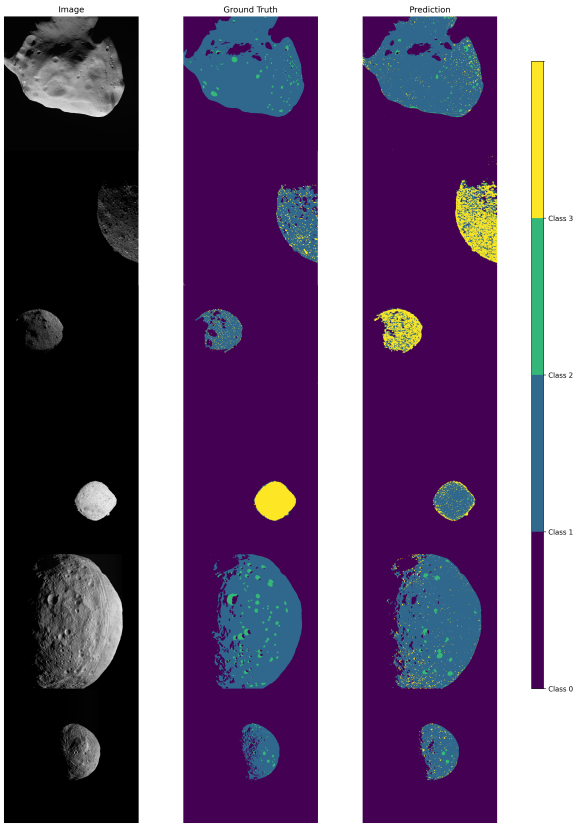


Figure 22: MultiRes U-Net mosaic

6. Conclusions and Future Works

In this work, an improved procedural framework for synthetic dataset generation for semantic segmentation of small bodies is presented. All the steps for procedural generation of asteroids, using Blender software, are described in detail, from the assembly of the 3D model, passing trough the material selection, to the automatic labeling strategy and the exportation of image-masks pairs. The proposed framework improves both efficiency and flexibility of the previous approach by introducing the following innovations:

- Exploiting Python integration in Blender, craters can be automatically sculpted on the mesh, removing the need for manual placement. This allows the realistic simulation of a large number of tiny craters, reduces the model weight, and saves significant time.
- By combining Material Index and Vertex Groups, all masks can be extracted from a single model, without the need of saving different version of the model for each mask. Vertex Groups also allow direct mesh-level segmentation, introducing a halfway approach between purely synthetic labeling and fully manual annotation.
- By exploiting Light Passes, background/shadow masks can be automatically extracted without the need of external post processing for computing terminator region masks.

A noise analysis is conducted on real imagery and, from the extracted noise, an equivalent model is built with the aim of improve the networks robustness and the segmentation performance on real asteroid images, therefore two noise augmentation strategies are implemented. The procedural framework is exploited to generate synthetic datasets for training of CNNs. Three U-Net models and MultiRes U-Net are trained and evaluated in three test cases, with different noise augmentation strategies, and under domain shift condition on a set of manually labeled real mission imagery, employing Accuracy, Mean Intersection over Union and per-class Intersection over Union as metrics of comparison. The obtained results are then compared with the outcome of the work of M.Pugliatti and M.Maestrini [19]:

- Surface, craters and boulders classes IoU are aligned with the previous works
- Background class IoU improved from 70% to 98%, together with MIoU, from 39% to 48%, most probably due to the use of a different mask that better fit this the segmentation task

In the context of synthetic-data-only training, further improvements can be achieved by ensuring better balance

in the training datasets, also by an optimization of classes weights, and by employing appropriate weight initialization or pretrained models instead of training from scratch. In addition, auxiliary tasks could be introduced to provide the network with information about surface concavity, supporting the discrimination between craters and boulders.

While these aspects address the data and training setup, the choice of architecture also plays a crucial role in determining the segmentation performance. In this regard, the MultiRes U-Net architecture demonstrated superior performance compared to the baseline U-Net across all classes, confirming its stronger generalization capability under domain shift. Nevertheless, its potential has not been fully exploited due to the limitations of the training setup. Future work should therefore focus on:

- Designing better training strategies to leverage the capacity of MultiRes U-Net.
- Exploring transformer-based segmentation architectures which are expected to further improve performance.
- Investigating domain adaptation strategies specifically tailored to the asteroid segmentation task, to reduce the gap between synthetic and real domains.

References

[1] OSL Language: https://open-shading-language.readthedocs.io/en/v1.13.12.0/ (accessed:29/09/25).

[2] 3D Asteroid Catalogue: https://3d-asteroids.space (accessed:29/09/25).

[3] pds3 Archive: https://sbnarchive.psi.edu/pds3/ (accessed:29/09/25).

[4] Chirold D. Epp, Edward A. Robertson, and Tye Brady. Autonomous landing and hazard avoidance technology (alhat). pages 1–7, 2008.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. 2016. http://www.deeplearningbook.org.

[6] B. Hapke. Bidirectional reflectance spectroscopy 3. correction for macroscopic roughness. *Icarus*, 59:41–59, 07 1984.

[7] Bruce Hapke. Theory of reflectance and emittance spectroscopy. 2012.

[8] P.H. Hasselmann, M.A. Barucci, S. Fornasier, C. Leyrat, J.M. Carvano, D. Lazzaro, and H. Sierks. Asteroid (21) lutetia: Disk-resolved photometric analysis of baetica region. *Icarus*, 267:135–153, 2016.

[9] Nabil Ibtehaz and M. Sohel Rahman. Multiresunet : Rethinking the u-net architecture for multimodal biomedical image segmentation. *Neural Networks*, 121:74–87, 2020.

[10] Paolo Lunghi, Marco Ciarambino, and Michèle Lavagna. Vision-based hazard detection with artificial neural networks for autonomous planetary landing. 05 2015.

[11] Raymond E. March. Osiris-rex: a nasa asteroid space mission. *International Journal of Mass Spectrometry*, 469:116677, 2021.

[12] N. Masoumzadeh, H. Boehnhardt, Jian-Yang Li, and J.-B. Vincent. Photometric analysis of asteroid (21) lutetia from rosetta-osiris images. *Icarus*, 257:239–250, 2015.

[13] Rahul Moghe and Renato Zanetti. A deep learning approach to hazard detection for autonomous lunar landing. *The Journal of the Astronautical Sciences*, 67:1–20, 10 2020.

[14] Michael Oren and Shree K. Nayar. Generalization of lambert's reflectance model. page 239–246, 1994.

[15] PANGU: https://pangu.software (accessed:29/09/25).

[16] POV-Ray: http://www.povray.org (accessed:29/09/25).

[17] Ravi Prakash and Karmanya Kumar. Class weight technique for handling class imbalance. 07 2022.

[18] L. Prockter, S. Murchie, A. Cheng, S. Krimigis, R. Farquhar, A. Santo, and J. Trombka. The near shoemaker mission to asteroid 433 eros. *Acta Astronautica*, 51(1):491–500, 2002.

[19] Mattia Pugliatti and Michele Maestrini. Small-body segmentation based on morphological features with a u-shaped network architecture. *Journal of Spacecraft and Rockets*, 59:1–15, 09 2022.

[20] https://keras.io/api/callbacks/reduce_lr_on_plateau/ (accessed:29/09/25).

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. 2015.

[22] C.T. Russell, A. Coradini, W.C. Feldman, Ralf Jaumann, A.S. Konopliv, Thomas Mccord, Lucy Mcfadden, Harry McSween, S. Mottola, G. Neukum, C.M. Pieters, C. Raymond, David Smith, Mark Sykes, Bobby Williams, and M.T. Zuber. Dawn : A journey to the beginning of the solar system. pages 63–66, 01 2002.

[23] Kento Tomita, Katherine Skinner, Keidai Iiyama, Bhavi Jagatia, Tatsuwaki Nakagawa, and Koki Ho. Hazard detection algorithm for planetary landing using semantic segmentation. 11 2020.

[24] Yuichi Tsuda, Satoru Nakazawa, Makoto Yoshikawa, Takanao Saiki, Fuyuto Terui, Masahiko Arakawa, Masanao Abe, Kohei Kitazato, Seiji Sugita, Shogo Tachibana, Noriyuki Namiki, Satoshi Tanaka, Tatsuaki Okada, Hitoshi Ikeda, and Sei ichiro Watanabe. Chapter 2 - mission objectives, planning, and achievements of hayabusa2. In Masatoshi Hirabayashi and Yuichi Tsuda, editors, *Hayabusa2 Asteroid Sample Return Mission*, pages 5–23. Elsevier, 2022.

[25] Jacopo Villa, Jay Mcmahon, and Issa Nesnas. Image rendering and terrain generation of planetary surfaces using source-available tools. 02 2023.

[26] P. Villefranche, J. Evans, and F. Faye. Rosetta: The esa comet rendezvous mission. *Acta Astronautica*, 40(12):871–877, 1997.

[27] Makoto Yoshikawa, Junichiro Kawaguchi, Akira Fujiwara, and Akira Tsuchiyama. Chapter 6 - the hayabusa mission. In Andrea Longobardo, editor, *Sample Return Missions*, pages 123–146. Elsevier, 2021.

A. Appendix A

A.1. Hapke 5 parameters reflectance model

The bidirectional reflectance $r(i, e, g)$ according to the Hapke model is expressed as:

$$r(i,e,g) = \frac{w}{4\pi}\,\frac{\mu_0}{\mu_0+\mu}\left[\left(1+B(g)\right)P(g) + H(\mu_0)\,H(\mu) - 1\right]S(\bar{\theta}) \tag{10}$$

where:

- i is the incidence angle, e is the emission angle, and g is the phase angle,
- $\mu_0 = \cos i$, $\mu = \cos e$,
- w is the single-scattering albedo,
- $P(g)$ is the particle phase function,
- $B(g)$ is the opposition effect function (SHOE),
- $H(\mu)$ is the multiple-scattering (Hapke) function,
- $S(\bar{\theta})$ is the macroscopic roughness correction for mean slope $\bar{\theta}$.

$P(g)$ is the single-term Henyey-Greenstein (HG) phase function:

$$P(g) \;=\; \frac{1-g^2}{\left(1+2g\cos g+g^2\right)^{3/2}}, \tag{11}$$

where the parameter $g \in (-1, 1)$ controls the asymmetry of scattering: $g < 0$ is backscattering, $g > 0$ is forward scattering, and $g = 0$ is isotropic scattering.
The SHOE term is commonly expressed as:

$$B(g) \;=\; \frac{B_0}{1+\frac{1}{h}\tan\left(\frac{g}{2}\right)}, \tag{12}$$

where B_0 is the amplitude of the opposition surge and h is its angular half-width.
The factor $S(\bar{\theta})$ in (10) accounts for macroscopic surface roughness (average slope $\bar{\theta}$) and modifies the reflectance to include facet shadowing and geometric effects due to non-planar surface elements. The full expression for the roughness correction is given in Hapke's formulation; here we denote it compactly as $S(\bar{\theta})$.
For completeness, refer to Hapke for the full explicit form of the roughness correction:

$$S(\bar{\theta}) = S(i, e, g; \bar{\theta})$$

Abstract in lingua italiana

L'esplorazione degli Small Body, come asteroidi e comete, richiede un elevato grado di autonomia da parte della sonda spaziale per operazioni critiche quali l'atterraggio e l'Hazard Detection and Avoidance (HDA), in cui la segmentazione di immagini riveste un ruolo fondamentale. Le principali sfide in questo ambito derivano dalla scarsità di immagini annotate di missioni reali, e dal divario presente fra i dati sintetici e dati reali. Questa tesi affronta tali problematiche proponendosi di migliorare il framework procedurale esistente in letteratura per la generazione di dataset sintetici con annotazione automatica. Vengono esplorate strategie di noise augmentation per mitigare il domain shift, e due architetture di rete convoluzionale, una rete U-Net come riferimento e una più complessa MultiRes U-Net, sono addestrate esclusivamente su dati sintetici e testate su immagini reali etichettate manualmente. I risultati sono in linea con i lavori precedenti di M.Pugliatti e M.Maestrini, ma mostrano un netto miglioramento nella segmentazione dello sfondo e un incremento della Mean Intersection over Union (MIoU). MultiRes U-Net ha ottenuto prestazioni migliori rispetto alla rete di riferimento, dimostrando maggiori capacità di generalizzazione in condizioni di domain shift. Lo studio mette in evidenza l'importanza della preparazione del dataset di addestramento, e la necessità di sfruttare architetture e strategie di allenamento più avanzate per ridurre ulteriormente il divario tra dominio sintetitco e reale a supporto dell'esplorazione spaziale autonoma.

Parole chiave: Asteroidi, Segmentazione Semantica, Dati Sintetici, Arricchimento con Rumore, Reti Neurali