CNN-BASED AUTONOMOUS HAZARD DETECTION: A LIDAR-LESS APPROACH.

Pelayo Peñarroya[(1)], **Pablo Hermosín**[(1)], **Simone Centuori**[(1)]

[(1)] *Deimos Space S.L.U., Ronda de Poniente 19, Tres Cantos (Madrid), 28760, Spain*

ABSTRACT

Hazard Detection and Avoidance (HDA) systems typically rely on Light Detecting And Ranging (LIDAR) sensors to combine the visual observations with ranging measurements that help understand the steepness and roughness of the terrain underneath the spacecraft. However, this sensor suite is usually expensive, heavy, and takes up a considerable volume in the spacecraft platform. In this work, a LIDAR-less method is proposed, where Convolutional Neural Networks (CNNs) are trained and used to detect shadows, features, and sloped terrain. Using only passive instruments such as cameras (as opposed to active instruments such as LIDARs) is a very challenging problem and makes assessing the topology of the surface of interest the key to the proposed approach.
To cope with that, a dataset is generated using Blender that includes several small celestial body geometries to increase the generalisation capabilities of the networks. Stochastic feature populations are distributed on the surface of the different meshes used to add variability and enrich the dataset, and different illumination conditions are also considered. The networks trained are based on the classic Residual Network (ResNet) architecture and exploit the benefits transfer learning offers (in particular for semantic segmentation problems like the one described here) by initialising their weights to those obtained from the ImageNet dataset.
Results show that the networks are capable of predicting very accurately the hazards present in the input observations provided, particularly for shadows and features. Performances for slope estimation show an overall accuracy always above 75%, which is high for this type of problem. Furthermore, the composition of the three layers trained (feature detection, shadow detection, and slope estimation) show that the safety maps generated are capable of very accurately predicting and deciding where a landing can safely take place. Preliminary Hardware In the Loop (HIL) tests are also included to show the algorithm's performances when real hardware is used for image acquisition.

1 INTRODUCTION

Interacting with planetary surfaces is becoming a key technology for the current space market. Missions like Origins, Spectral Interpretation, Resource Identification, and Security-Regolith Explorer (OSIRIS-REx), Hayabusa2, or Double Asteroid Redirection Test (DART) are clear examples that the goal when flying a spacecraft in a small-body environment is to take a sample of the surface materials to take back to Earth for further analysis or to make contact with the asteroid or comet in some way; even to the point of trying to deviate from its trajectory. Be that as it may, whether it is for scientific or for planetary defence purposes (and maybe soon for mining purposes too), the capability to interact with a planetary body is definitely something that any mission to a small celestial body would benefit from having.

However, interacting with a planetary surface is a very complex problem that involves many other disciplines, such as relative navigation, trajectory design and control, contact dynamics analysis, risk management, or autonomous operations; among others.
The methods involved in assessing whether a certain surface is a safe or dangerous spot to land are often used in HDA algorithms. HDA systems and algorithms are used to identify and avoid elements that could be considered potentially risky, such as boulders, cracks, craters, holes, steep terrain, or shadowed areas, for instance. The avoidance is usually taken care of by the guidance and control modules of the spacecraft, while the detection is tackled by the navigation module. HDA systems have been investigated since [1, 2], where their use for lunar landing was discussed.
HDA systems typically rely on LIDAR sensors to combine the visual observations with ranging measurements that help understand the steepness and roughness of the terrain underneath the spacecraft. However, this sensor suite is usually expensive, heavy, and takes up a considerable volume in the spacecraft platform. In fact, they could weigh up to $25\,\mathrm{kg}$ and require up to $200\,\mathrm{W}$ of power [3]. LIDARs are considered active instruments because they are actively interacting with the environment to take observations, i.e., they emit a laser beam that is afterwards received by the instrument itself to measure the way in which the beam was reflected by the surface. The above-mentioned works in [1, 2] are based on this type of active sensors.
On the other hand, optical sensors such as cameras, on the other hand, are considered passive instruments because they only take information from the environment, without actively interacting with it. Using only passive instruments is a very challenging problem because they offer no range-like observation, which is the type that is typically exploited to build Digital Elevation Maps (DEMs). This makes assessing the topology of the surface of interest the key element of the proposed approach. Passive systems have also been discussed in [3] or [4], but it was concluded that the slope estimation was lacking.
The key element that these technologies were missing was more powerful and resource-efficient Image Processing (IP) capabilities. Until a few years ago, traditional IP techniques were used, which are challenging because of their computational burden and hardware limitations. Currently, Artificial Intelligence (AI) offers new tools to work around this problem, mainly in the form of Artificial Neural Network (ANN). In [5], it is proposed to use these to tackle navigation scenarios. ANNs drastically reduce the computational effort needed to process images, rendering them very appealing for on-board applications. CNNs in particular are a type of network that use convolutions to identify critical features in an image. In [6], this architecture was studied and used to identify lunar craters, showing very promising transferability and generalisation properties by training a network based on lunar images and testing the networks on images of Mercury's surface.
In more current pieces of work, CNNs have shown to be the generally preferred architecture for IP tasks. In [7, 8], they are used to estimate the pose of a non-cooperative spacecraft. Then, in [9], it was concluded that CNN-based algorithms perform better than traditional techniques. More recently, [10, 11] have shown how these networks perform for orbital navigation scenarios.
However, one of the main drawbacks of Deep Learning (DL) architectures such as CNNs is the large amount of data they need [12, 13]. Moreover, because the training scheme is based on a supervised approach, the data has to be labelled.
Even with all of the above, developing algorithms to process passively acquired images that are able to assess the geometry and roughness of the observed body still remains an important problem to tackle. To cope with that, the proposed method exploits the shadows cast by the geometry of the planetary body to infer the slopes in the terrain, relative to the local gravity vector at the surface.
In this work, a method is proposed that attempts to solve some of the most troublesome aspects of the detection part of the system: hardware requirements and cost. A focus is placed on the role of

autonomous navigation and, in particular, on the identification of elements that could put to risk a potential landing or interaction with the orbited body. This paper presents a LIDAR-less method, where CNNs are trained and used to detect features, shadows, and sloped terrain.
In the following, section 2 will introduce the architectures used for the networks and the techniques used to develop them. Section 3 will describe how the artificial datasets used for the training were generated. After that, section 4 will show the results obtained after training and will include some initial estimates for images taken by real sensors. To close out, section 5 sums up the key aspects of this work and proposes some new lines of work for the future.

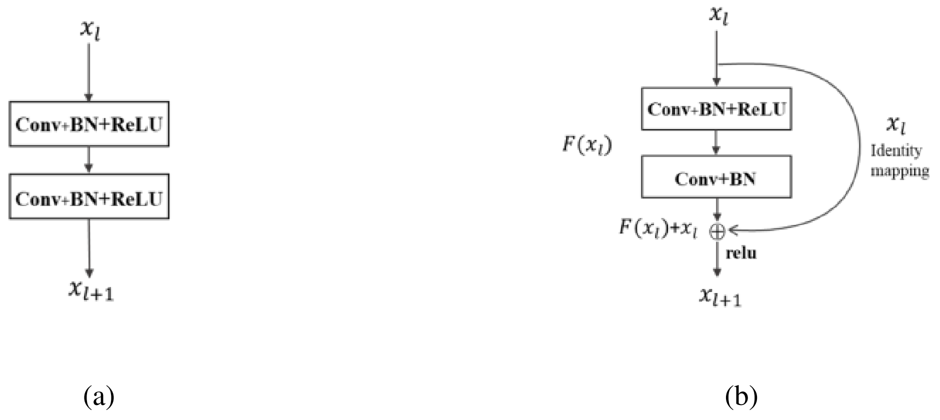


Figure 1: Comparison between a traditional convolutional block (left) and a residual block (right).

2 ARCHITECTURE

CNNs are multilayer feed-forward structures that use convolutions to apply image processing filters. Stacking several layers of these filters achieves very accurate pixel-by-pixel classifications. Three main types of layers are present in these architectures:

1. Convolutional layers: image filters are applied to identify a certain pattern or feature.
2. Pooling layers: feature maps are reduced to their main attributes.
3. Fully-connected layers: final classification.

Training a network from zero is quite a cumbersome process that requires a great deal of computational power and resources. Thus, transfer learning [14] is used to start training the networks from a state that has already been tested and validated. This reduces the computational burden of the training and allows to add more flexibility to the training by minimising the effort needed to get accurate network predictions.
Another negative aspect of CNNs is what is called the vanishing gradients problem. To train a neural network, the partial derivatives at each layer are needed to propagate upwards the weight modifications that are inferred from the comparison between the current estimate and the labelled ground truth. When the network architecture becomes too deep, i.e. too many layers are stacked, these derivatives become smaller for each step towards the first layers that they take, not affecting those initial layers at all. In [15], a novel solution was proposed where residual layers were introduced to prevent that loss of information with depth, as depicted in figure 1. By doing so, deeper architectures were possible to

train, which are able to obtain a much more complete understanding of the different features present in the image. These are called ResNets.
The networks used in this work are based on the classic ResNet architecture (in particular ResNet-34, see Table 1) and exploit the benefits that transfer learning offers (especially for semantic segmentation problems like the one described here) by initialising its weights to those obtained from the ImageNet dataset.

Table 1: ResNet-34 architecture.

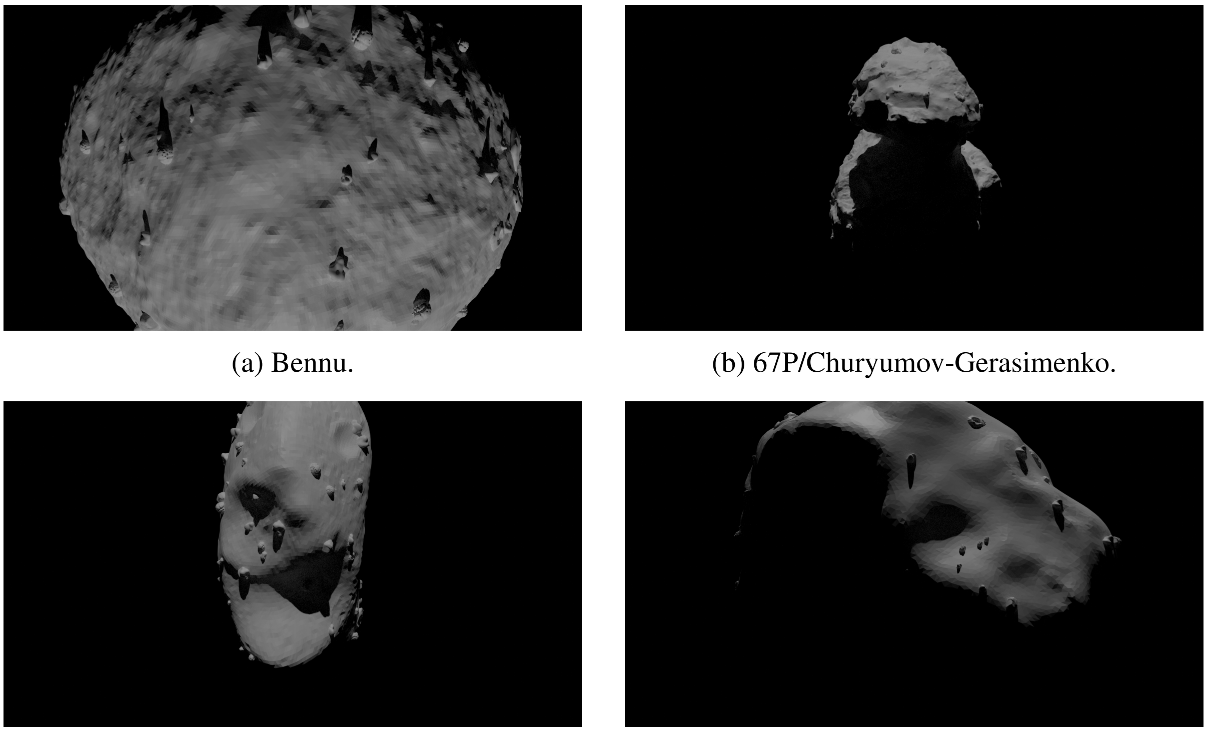
Layer	Output size	34-Layer
Conv1	112×112	$7 \times 7, 64, stride2$ $3 \times 3 maxpool, stride2$
Conv2-x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$
Conv3-x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$
Conv4-x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$
Conv5-x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$
	1×1	Average pool, 1000-d, softmax

3 DATASET GENERATION

Following recommendations by [16, 17], three datasets are usually needed to train a neural network and bring it to an operational stage:

- Training set: the largest set of the three. Used to train the weights of the network.
- Validation set: a smaller set used during the training to keep the neural network from over-fitting to the training set.
- Test set: another small set not used in the training, used to validate the operational use of the network.

To train the networks intensively, a dataset is generated using Blender that includes several small celestial body geometries to increase their generalisation capabilities. Stochastic feature populations are distributed on the surface of the different meshes used to add variability and enrich the dataset, and different illumination conditions are also considered. Using several randomly generated initial orbital states, images are rendered from very different orbital positions and orientations, to broaden the different viewpoints from which predictions are expected. Figure 2 exemplifies the different meshes used for the training dataset.
As has been introduced before, the training scheme follows a supervised approach. Thus, a set of true labels needs to be provided, along with the rendered images used for prediction. The labels proposed for this task are four: *safe*, *unsafe*, *deep-space*, and *undetected*. Table 2 gathers the different labels included, their colouring, and their meaning.
Since there are three layers that need to be trained for, each of the rendered images needs to be paired up with three masks, namely feature detection, shadow detection, and slope estimation. Figure 3


(a) Bennu. (b) 67P/Churyumov-Gerasimenko.

(c) Eros. (d) Lutetia.

Figure 2: Examples of the different base meshes and feature distributions for the training and validation dataset.

Table 2: Label meaning for each of the layers developed for *astroHda*.

	Shadow detection	**Feature detection**	**Slope estimation**	**Color**
Safe	Illuminated	Non-featured	Slope $< 15^\circ$	Green
Unsafe	Shadowed	Featured	Slope $\geq 15^\circ$	Red
Deep-space				Blue
Undetected				Black

shows the truth masks for an arbitrary image of 67P/Churyumov-Gerasimenko included in the training dataset.
Feature masks are generated by pixel-subtraction techniques comparing the featured mesh against the original (naked) mesh. Shadows are detected by pixel brightness and ruling out the deep-space background. Slopes are divided into *safe* and *unsafe* based on a threshold of 15°, which is based on [18]. To generate those masks, the slope of each of the facelets of the shape model used for the rendering is computed with respect to its local gravity vector and labelling them as *safe* if the value is under 15° and *unsafe* otherwise. Figure 4 provides an example of how the 67P/Churyumov-Gerasimenko sloped mesh looks.
In total, 2000 images were generated for the dataset, to which augmentation techniques were applied during training. The augmentation parameters are gathered in Table 3. The images in the dataset are split for training and validation in an 80-20 ratio and the rendered images are resized to 300×300 pixels. The training is performed in two stages: an initial one were only the last layers are free, preventing the weights acquired using transfer learning to change in the first layers of the network and thus conserving the high-level feature detection capabilities; and a final stage were the last layers

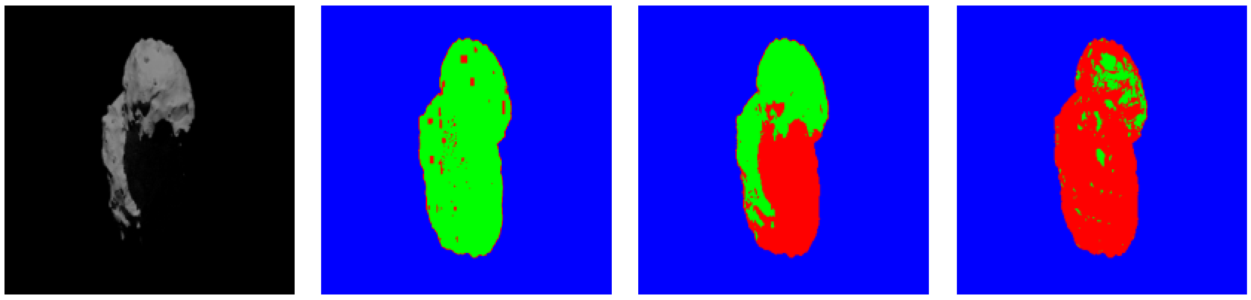


Figure 3: Examples of the masks generated for an arbitrary rendering of 67P/Churyumov-Gerasimenko. From left to right: original rendered image, features, shadows, slopes. Blue represents deep-space, red represents dangerous areas, and green represents safe areas.

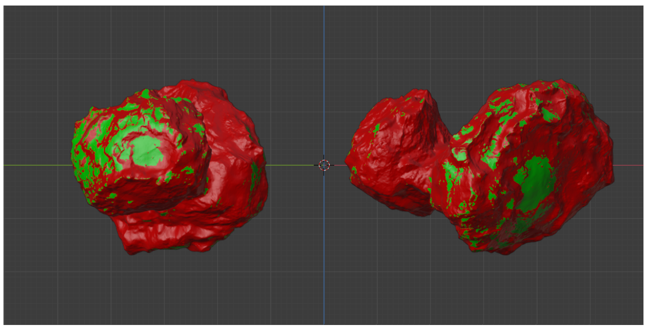


Figure 4: 67P/Churyumov-Gerasimenko shape model with its facelets coloured according to the slope threshold established at $15°$. Green and red faces correspond to areas with slopes lower or higher than $15°$, respectively.

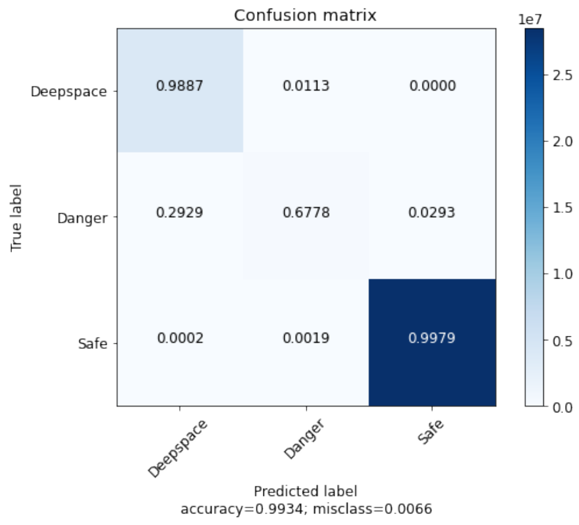
are unfrozen to fine-tune the last layers to the nature of the images used. In total, 22 epochs of training were used (10 for the initial stage and 12 for the final one), where an Intersection over Union (IoU)-based method was used as accuracy metric and a FlattenedLoss of CrossEntropy-Loss is used as loss-function.

Table 3: Augmentation parameters for training dataset

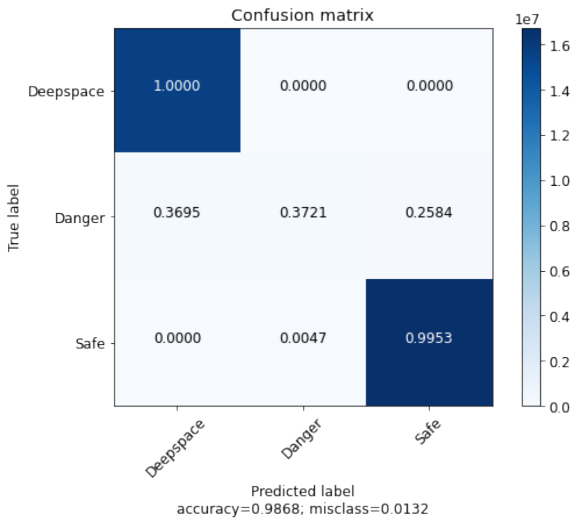
Random flip	True
Max rotation angle	10.0 deg
Min/Max zoom	1.0 / 5.0
Max lightning change	0.2
Max warping	0.2
Probability of affine transformation	0.75
Probability of brightness change	0.75
Gaussian noise mean/σ	0.0 / 10.0

4 RESULTS

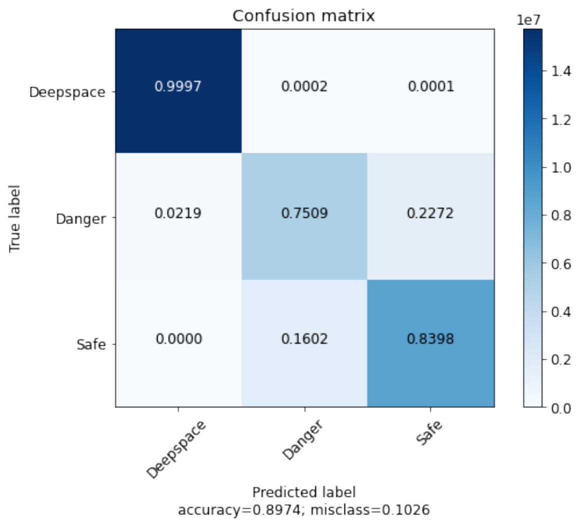
With the networks trained, the quality of the predictions can be assessed by comparing them against the true labels. On a pixel-per-pixel basis, as it is typically done for semantic segmentation tasks, the percentages of True Positives (TP) and False Positives (FP) are computed, based on training statistics. These numbers are provided in figure 5.



(a) Shadow detection.



(b) Feature detection.



(c) Slope estimation.

Figure 5: Statistics obtained from the training dataset. Performances are normalised with respect to the total number of true labels for each category, shown in the color bar.

The confusion matrices for the training set show that the predictions performed by the networks are very high for the *deepspace* and *safe* labels, while slightly lower for the *danger* one. There are two reasons for that: the first one being the performances of the network themselves, and the second being related to the labelling algorithm.
For the shadow detection layer, safe (illuminated) areas are very accurately identified, with the amount of FP for that category being around a 0.2% of the totality of *safe* labels. Deep space is also accurately identified, providing the additional capability of limb detection. The detection of the shadows seems to be slightly blurred by the presence of deep space close to shadowed areas, which makes it difficult for the network to estimate the limits between these two conditions, i.e., to estimate the limb of the body when illumination conditions are very poor.

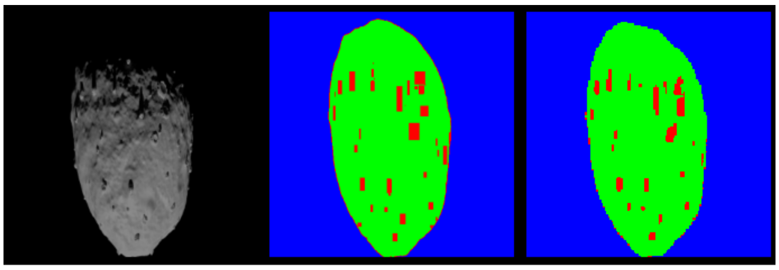


Figure 6: Arbitrary rendered image of Bennu (left), with its corresponding generated true feature mask (centre), and the prediction by the feature detection layer.

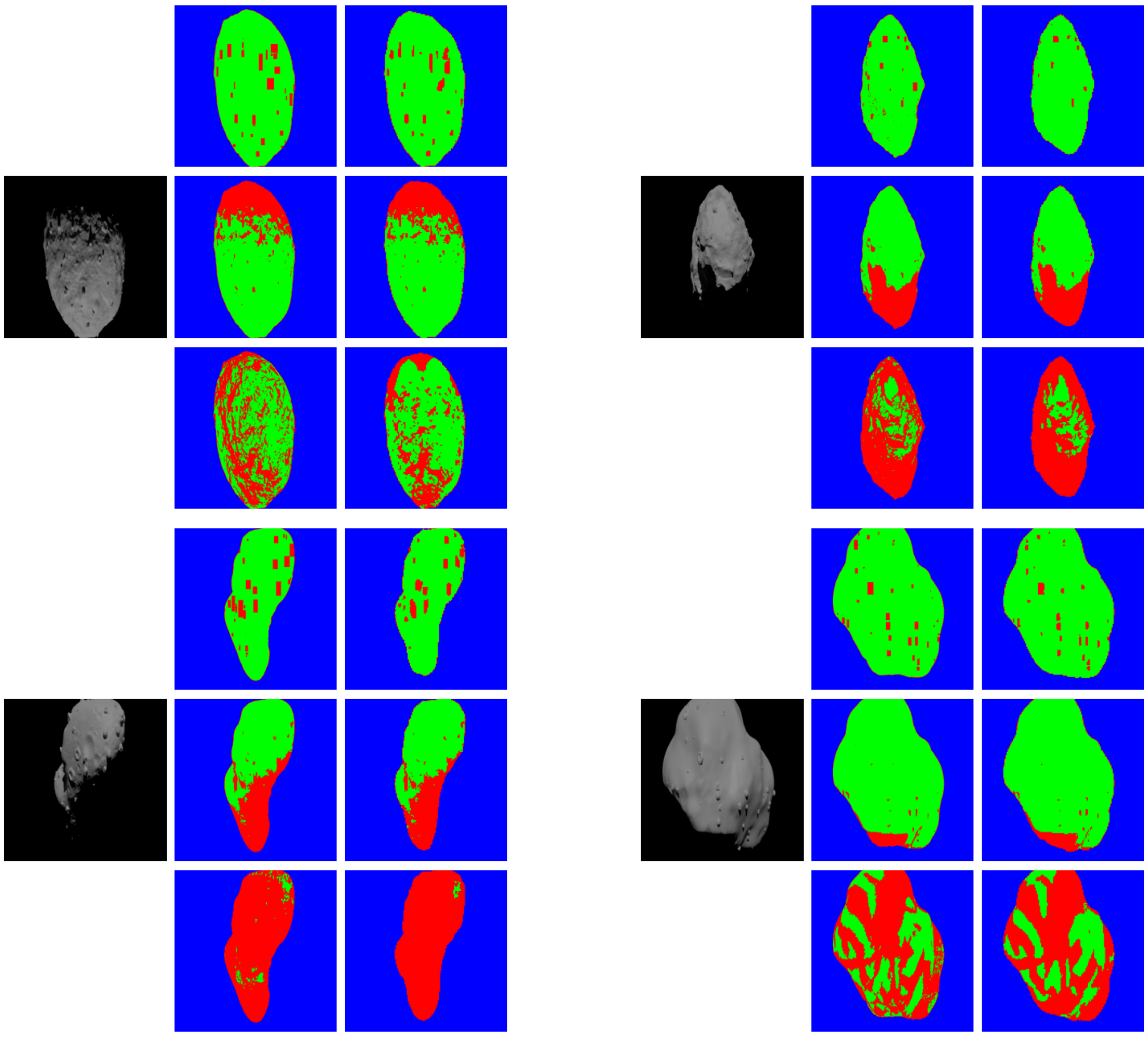


Figure 7: Test dataset results on four different rendered images. On each of the panels, the left-hand side corresponds to the rendered image given as input to the networks, the first column is the target, and the second column is the prediction. The first row corresponds to feature detection, the second to shadow detection, and the third to slope estimation.

Results for the feature detection layer show very similar behaviour to those of the shadow detection layer, even showing a greater amount of FP for the *unsafe* label. In this case, however, this is due to the labelling algorithm placing rectangles on top of the features to be detected. Figure 6 shows an example where these rectangles can be observed. In the prediction shown on the right-hand side of that figure, it can be observed how the network has learnt not only to detect the features but also to adapt the predictions to the natural shapes of the features, flagging features that resemble reality much more than the automatically placed rectangles. This explains why some of the *unsafe* labels within those rectangles are predicted as something else, raising the number of FP in that category.
The masks for the slope estimation layer, differently from the ones generated for the shadow and feature detection, are directly based on the geometry of the body and not on image processing techniques. This means that the accuracies shown in its confusion matrix are entirely due to the performance of the algorithm. The fact that similar amounts of FPs are observed for both the danger and safe categories confirms it. With a 75.09% of true positives for danger detection and an 83.98% for safe, the network still shows very accurate predictions, even in the layer that was anticipated to be most problematic, due to the lack of depth information in the inputs to the network of the LIDAR-less approach proposed in this work. Results for four images of the training dataset are provided in figure 7

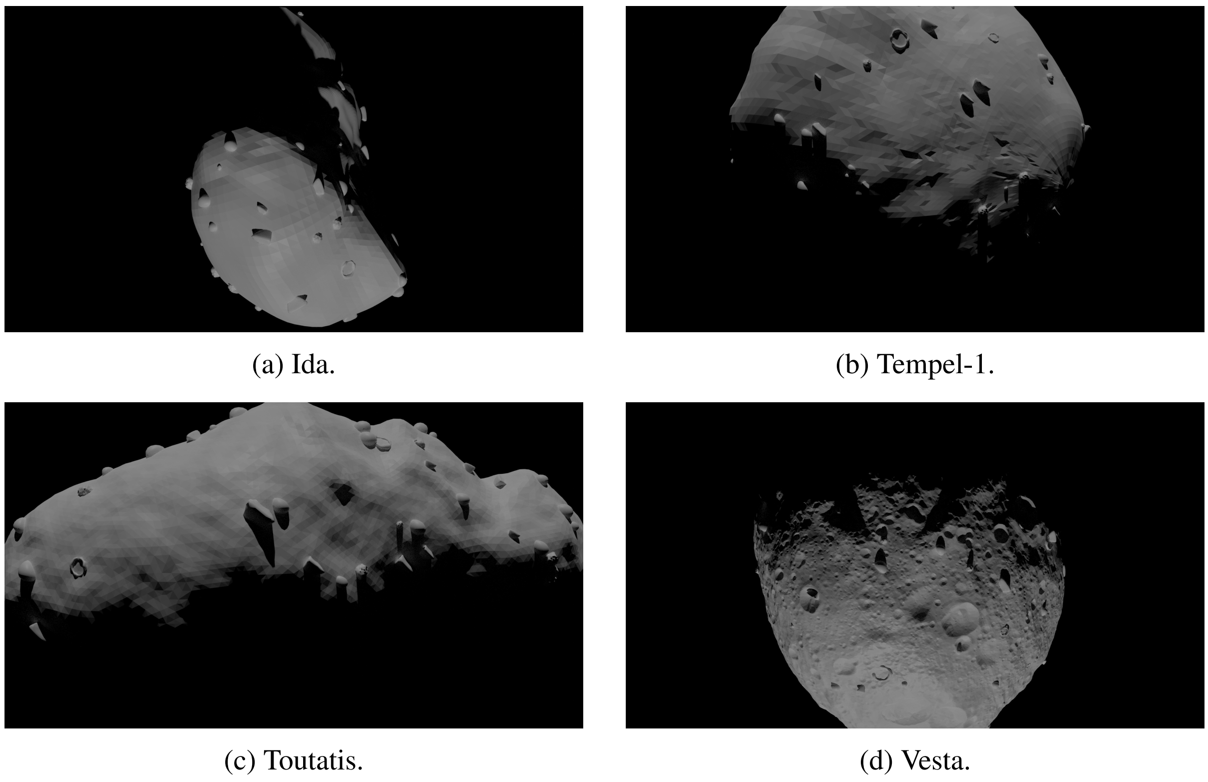
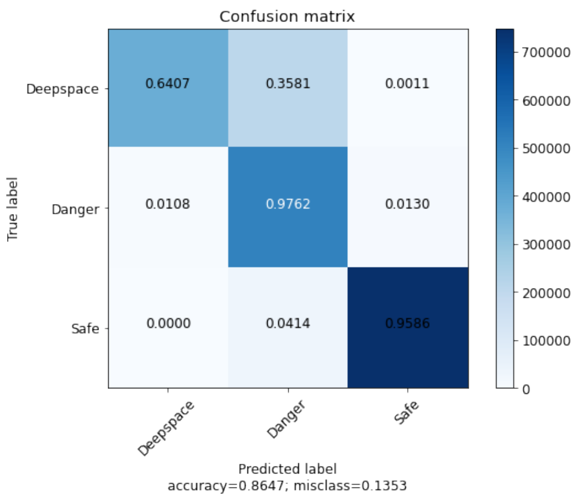

(a) Ida. (b) Tempel-1. (c) Toutatis. (d) Vesta.

Figure 8: Examples of the different base meshes and feature distributions for the test dataset.

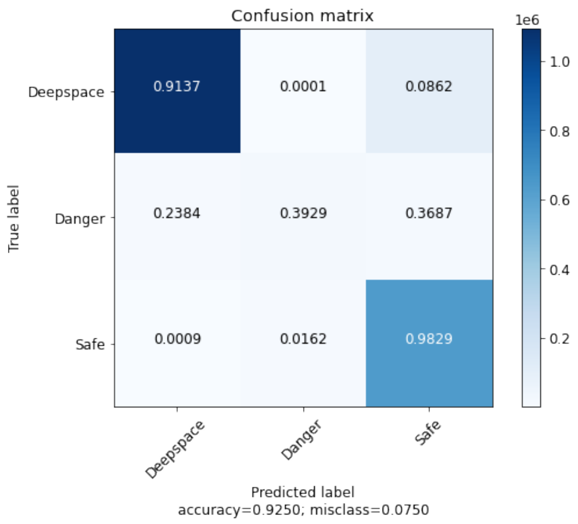
4.1 Test Set

One of the aspects of the proposed solution for HDA systems is that it needs to be able to generalise its predictions to (ideally) any body, which is the reason why different geometries and feature populations were included in the training dataset. To analyse whether this is the case for the developed networks, an additional dataset is created: the test set. This dataset has a size of 100 images and uses bodies, features, and feature distributions that are completely different to those used for the training dataset,

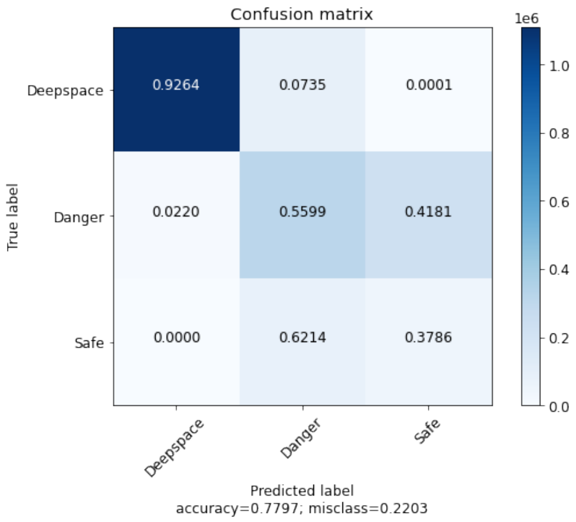
to avoid any possible adulteration on the predictions. Figure 8 shows examples of the images used in the test set, featuring Ida, Tempel-1, Toutatis, and Vesta as minor celestial bodies.
The statistics obtained indicate that the feature and shadow detection capabilities of the algorithm are very versatile and adapt well to unknown environments. On the other hand, slope estimation seems to struggle when the geometries present in the images are not included in the training phase of the network, going from 89.7% in overall accuracy to 78.0%, as it is shown in figure 9.



(a) Shadow detection.



(b) Feature detection.



(c) Slope estimation.

Figure 9: Statistics obtained from the test dataset. Performances are normalised with respect to the total number of true labels for each category, shown in the color bar.

4.2 Safety Maps

The final outcome of this algorithm is the safety maps corresponding to the acquired images. Safety maps are generated by overlaying the predictions for each layer using their probability masks as deciding factors, i.e., integrating them together before a unique label is chosen for each of the layers.

Figure 10 provides an example of a safety map. In it, the probability maps that each layer of the network uses to choose the final label are shown. These probabilities (or how certain the network is about a certain pixel label) go from 0 to 1 and are the direct result of the convolutions and deconvolutions that take place in the U-shape architecture of the network.
To compose these probabilities and stack them together, a priority criterion is defined, in which deep space pixels rule over danger pixels, which in turn rule over safe pixels. This approach is taken to take a conservative perspective on the final labelling since it suffices that one of the three layers considers a pixel to be not safe to predict it as a hazard or deep space. Keeping the application in mind, this method achieves the main target, which is discarding any pixel that has not been identified as safe by all three layers, supporting thus the robustness of the algorithm before unclear areas of the minor celestial body.

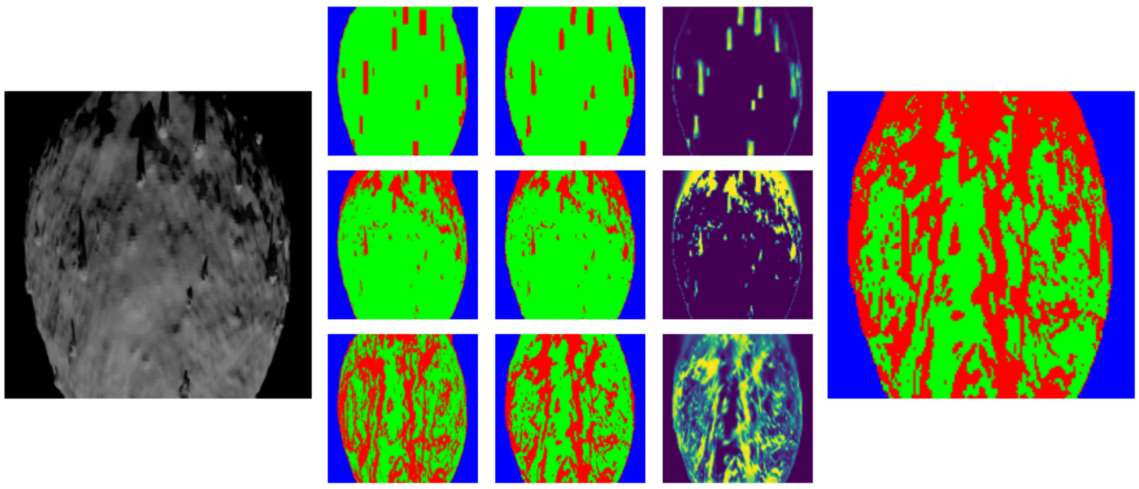


Figure 10: Example of a rendered image, the evaluation of the different hazard detection layers, and the final safety map composition. Left is the input rendered image, centre shows (from left to right) the target masks, the predictions, and the probability maps for feature detection, shadow detection, and slope estimation respectively; and right displays the final safety map.

4.3 Hardware-In-the-Loop

As explained before, one of the main drawbacks of Machine Learning (ML)-based methods is their low Technology Readiness Level (TRL) in space applications. In an effort to start the process of taking this work to its next stage, a preliminary HIL campaign was performed at Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) Bremen, where facilities to imitate small body illumination conditions. The setup used can be observed in figure 11, where the left-hand image shows the features available at the facilities, plus the 3D-printed mount onto which a Raspberry Pi 4 model B[1] was attached. The tests performed during this campaign executed the trained networks directly from the Raspberry Pi, which was capable of generating the safety maps shown before in around 10 seconds. Since the implementation of the networks on the Raspberry Pi was not specifically tailored, and no dedicated hardware was included to deal with CNN-based architectures, no data about the processor was collected at this stage.
The right-hand side of figure 11 shows the lights used to simulate small celestial body illumination

[1]Specifications for the Raspberry Pi 4 Model B can be found here.

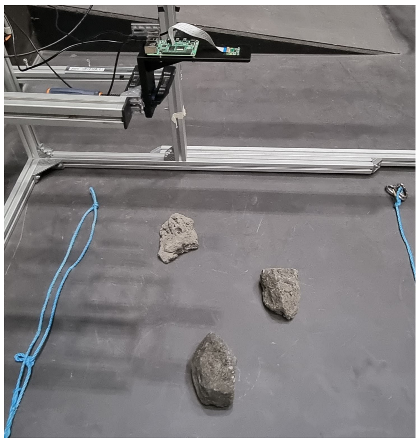
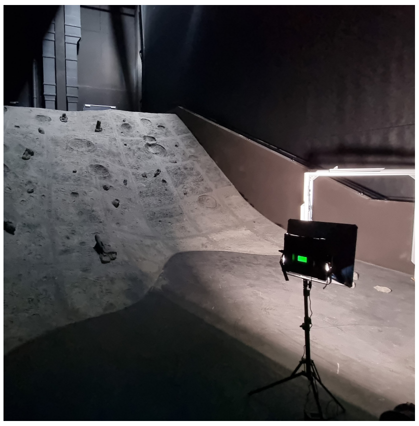



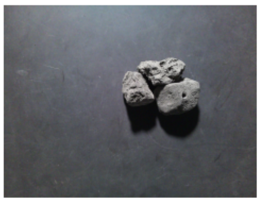
Figure 11: The setup used at DFKI Bremen for the image acquisition. On the left, a close-up look at the mount used to hold together the camera and the processing unit. On the right, an overview of the Space Hall before completely darkening the room.

conditions. A sample of the images taken using a Raspberry Pi Camera Module v2[2] can be seen in figure 12. There, different illumination angles and intensities are used. The position and number of elements are also varied to test the algorithm under a wider set of circumstances. Figure 13 shows the predicted features detected by the networks. The inclusion of Gaussian noise in the training proved to be fundamental for the processing of these real images, in which camera noise is present.

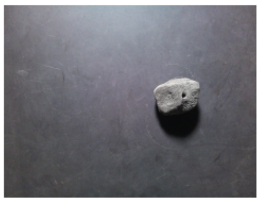
(a) Three elements in good illumination conditions.

(b) Three elements in poor illumination conditions.

(c) One element in good illumination conditions.



(d) Clustered elements in good illumination conditions.



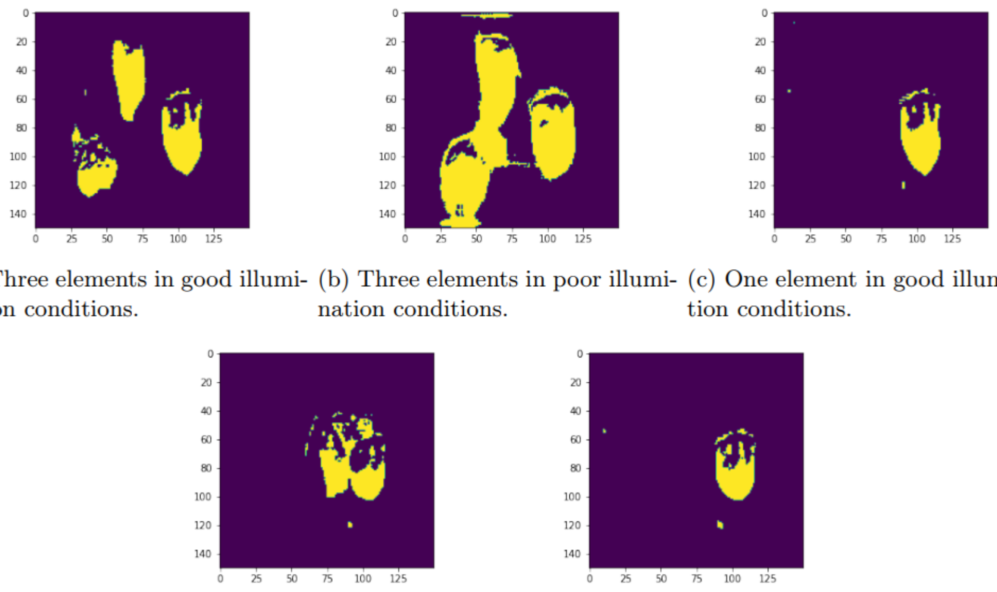
(e) Modified incidence angle for illumination conditions.

Figure 12: Images taken at DFKI, with different feature configurations and illumination conditions.

5 CONCLUSIONS

In this work, a CNN-based algorithm has been developed for HDA systems in which no LIDAR is required. This makes the whole system much lighter, smaller, and reduces its cost, from a component

[2]Specifications for the Raspberry Pi Camera Module v2 can be found here.


(a) Three elements in good illumination conditions. (b) Three elements in poor illumination conditions. (c) One element in good illumination conditions.

(d) Clustered elements in good illumination conditions. (e) Modified incidence angle for illumination conditions.

Figure 13: Predictions obtained for the different images taken at DFKI. Hazards are highlighted in yellow.

and a mission design perspective. Using only image observations as inputs is a very challenging task if information about the in-depth geometry of an object needs to be processed, however, CNNs prove to be capable of handling these scenarios by exploiting geometry-influenced factors such as shadows. A pipeline for dataset generation has been developed and used for the networks trained. This dataset generator provides versatility and eases the production of small celestial body environment images for the training of image processing networks. The generation of true masks to be used as labels for supervised training has also been developed, based on traditional image processing techniques and shape model geometry analysis. Some imperfections are found in the labelling system, which struggles to find a clear distinction between shadows and deep space and over-labels features due to the rectangles covering them being larger than the features themselves.
The chosen architecture (ResNet-34) shows great performances for semantic segmentation tasks, even with only a few training epochs, thanks to transfer learning. Augmentations help increase the variety and heterogeneity of the images in the dataset. With that training, three layers are generated for the prediction of shadows, features, and the estimation of slopes above a threshold of $15°$.
Including the shape models representing the actual geometries onto which the networks will be used improves the capabilities of the network to predict the limb of the body and helps with the slope estimation too, especially for areas that are not too well-lit.
Predictions obtained from the training and the test sets show overall accuracies always above 75%, being the slope estimation the worst of the layers. This was anticipated and, even if lower than the other layers (where accuracies always above 85% are achieved), it still represents a satisfactory benchmark.
Safety maps are generated using the probability maps predicted by the networks and using a priority-based methodology in which a conservative approach is chosen to predict as safe pixels only those that have been predicted as safe by all three layers.
Additionally, some preliminary HIL testing has been included where images were taken using a Rasp-

berry Pi camera module v2, mounted on a Raspberry Pi 4 model B at DFKI Bremen. Results obtained show that the inclusion of Gaussian noise is fundamental to cope with sensor noise.

6 ACKNOWLEDGMENTS

The authors of this paper 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.
Also, the authors thank DFKI Bremen for the use of their facilities and expertise to achieve some of the results shown in this paper.

7 REFERENCES

[1] C. D. Epp and T. B. Smith, "Autonomous Precision Landing and Hazard Detection and Avoidance Technology (ALHAT)," in *2007 IEEE Aerospace Conference*, Mar. 2007, pp. 1–7, iSSN: 1095-323X.

[2] A. E. Johnson, A. Huertas, R. A. Werner, and J. F. Montgomery, "Analysis of On-Board Hazard Detection and Avoidance for Safe Lunar Landing," in *2008 IEEE Aerospace Conference*. Big Sky, MT, USA: IEEE, Mar. 2008, pp. 1–9. [Online]. Available: http://ieeexplore.ieee.org/document/4526301/

[3] A. Huertas, Yang Cheng, and R. Madison, "Passive Imaging Based Multicue Hazard Detection for Spacecraft Safe Landing," in *2006 IEEE Aerospace Conference*. Big Sky, MT, USA: IEEE, 2006, pp. 1–14. [Online]. Available: http://ieeexplore.ieee.org/document/1655794/

[4] D. Neveu, G. Mercier, J.-F. Hamel, V. Simard Bilodeau, S. Woicke, M. Alger, and D. Beaudette, "Passive versus active hazard detection and avoidance systems," *CEAS Space J*, vol. 7, no. 2, pp. 159–185, Jun. 2015. [Online]. Available: http://link.springer.com/10.1007/s12567-015-0074-4

[5] P. Lunghi, M. Ciarambino, and M. Lavagna, "Vision-based hazard detection with artificial neural networks for autonomous planetary landing," in *13th ESA/ESTEC Symposium on Advanced Space Technologies in Robotics and Automation, ASTRA 2015*, 2015, pp. 1–8.

[6] A. Silburt, M. Ali-Dib, C. Zhu, A. Jackson, D. Valencia, Y. Kissin, D. Tamayo, and K. Menou, "Lunar Crater Identification via Deep Learning," *Icarus*, vol. 317, pp. 27–38, Jan. 2019, arXiv: 1803.02192. [Online]. Available: http://arxiv.org/abs/1803.02192

[7] L. Pasqualetto Cassinis, R. Fonod, E. Gill, I. Ahrns, and J. Gil Fernandez, “CNN-Based Pose Estimation System for Close-Proximity Operations Around Uncooperative Spacecraft,” in *AIAA Scitech 2020 Forum*. Orlando, FL: American Institute of Aeronautics and Astronautics, Jan. 2020. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2020-1457

[8] S. Sharma and S. D’Amico, “Neural Network-Based Pose Estimation for Noncooperative Spacecraft Rendezvous,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 6, pp. 4638–4658, Dec. 2020, conference Name: IEEE Transactions on Aerospace and Electronic Systems.

[9] K. Tomita, K. Skinner, K. Iiyama, B. Jagatia, T. Nakagawa, and K. Ho, “Hazard Detection Algorithm for Planetary Landing Using Semantic Segmentation,” in *ASCEND 2020*. Virtual Event: American Institute of Aeronautics and Astronautics, Nov. 2020.

[10] S. Silvestrini, M. Piccinin, G. Zanotti, A. Brandonisio, I. Bloise, L. Feruglio, P. Lunghi, M. Lavagna, and M. Varile, “Optical navigation for Lunar landing based on Convolutional Neural Network crater detector,” *Aerospace Science and Technology*, vol. 123, p. 107503, Apr. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1270963822001778

[11] M. Pugliatti, V. Franzese, and F. Topputo, “Data-driven Image Processing for Onboard Optical Navigation Around a Binary Asteroid,” *Journal of Spacecraft and Rockets*, vol. 59, no. 3, pp. 943–959, 2022, publisher: American Institute of Aeronautics and Astronautics.

[12] N. M. A.-M. M. Al-Moosawi and R. S. Khudeyer, “ResNet-34/DR: A Residual Convolutional Neural Network for the Diagnosis of Diabetic Retinopathy,” *Informatica*, vol. 45, no. 7, Dec. 2021, number: 7. [Online]. Available: https://www.informatica.si/index.php/informatica/article/view/3774

[13] M. Gao, J. Chen, H. Mu, and D. Qi, “A Transfer Residual Neural Network Based on ResNet-34 for Detection of Wood Knot Defects,” *Forests*, vol. 12, p. 212, Feb. 2021.

[14] S. Akçay, M. E. Kundegorski, M. Devereux, and T. P. Breckon, “Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery,” in *2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 1057–1061.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 2015, number: arXiv:1512.03385 arXiv:1512.03385 [cs]. [Online]. Available: http://arxiv.org/abs/1512.03385

[16] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge University Press, 2007, google-Books-ID: m12UR8QmLqoC.

[17] L. Ghilardi, A. Scorsoglio, and R. Furfaro, *ISS Monocular Depth Estimation Via Vision Transformer*, Sep. 2022.

[18] R. Wang, K. Di, W. Wan, Z. Liu, Y. Wang, W. Liang, X. Chen, and S. Zhi, “Topographic Mapping and Analysis Based on 3D Reconstruction Model of Simulated Asteroid,” *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B3-2020, pp. 1165–1170, 2020.