75th International Astronautical Congress (IAC), Milan, Italy, 14-18 October 2024.

IAC-24-C1.4.5

Integrated Optical Terrain Relative Navigation for Autonomous Lunar Landing

Giovanni Pio Parracino [a,*], **Michele Ceresoli** [a], **Stefano Silvestrini** [a], **Michèle Lavagna** [a]

[a] *Department of Aerospace Science and Technology, Politecnico di Milano, via La Masa, 34, Milano, 20134, MI, Italy,*
[*] *Corresponding author,* giovannipio.parracino@mail.polimi.it

Abstract

Spacecraft landing on celestial bodies is a very challenging phase for the Guidance, Navigation, and Control system. Due to the very fast dynamics, autonomous navigation to properly manage the whole landing phase in real-time is mandatory. A beneficial strategy in terms of both cost and complexity leverages vision-based navigation, as it only requires an on-board monocular camera.

In this regard, two main navigation strategies are leveraged: Relative Navigation (RN), which relies on features with unknown coordinates, but can only compute the relative pose with respect to the previous frame, and Absolute Navigation (AN), which instead allows determining the spacecraft pose with respect to a planetocentric reference frame exploiting features with known location, though asking for at least partial knowledge of the landing environment. Nonetheless, both methods have inherent limitations. AN experiences a decreasing number of known observable landmarks throughout descent, preventing state estimation during crucial final phases (altitudes below 15-20 km), while RN accumulates error over time, leading to a drift in the state estimation process.

This paper presents a novel navigation configuration to solve those limitations for pinpoint landing applied to the lunar specific scenario. An Integrated Navigation, based on the fusion of both information from Relative and Absolute Navigation, is adopted. In particular, the work focuses on the development of a suitable image processing pipeline for this integrated approach. Relative Navigation is here performed through Visual Odometry techniques. The features are extracted adopting an ORB detector with adaptive threshold and then tracked over multiple frames. Absolute Navigation adopts as landmarks craters which are located in the image using YOLOv7, an efficient state-of-the-art Object Detection Network. A method based on standard image processing techniques is developed to retrieve the elliptical crater rims from the detected bounding boxes, along with a robust crater matching strategy.

The algorithm performance and robustness under various illumination, camera pointing, and terrain conditions are evaluated through numerical simulations with synthetic lunar calibrated image. Benefits of the proposed image processing pipeline on the Integrated Navigation are discussed, together with the limits of the current implementation.

Acronyms

1NNS	1 Nearest-Neighbor Search.
AN	Absolute Navigation.
AT-ORB	Adaptive Threshold ORB.
CDMA	Crater Detection and Matching Algorithm.
CIoU	Complete IoU.
CNN	Convolutional Neural Network.
DEM	Digital Elevation Model.
FOV	Field Of View.
IEKF	Implicit Extended Kalman Filter.
IoU	Intersection over Union.
ML	Machine Learning.
MSAC	M-estimator Sample Consensus.
NMS	Non-Maximum Suppression.
ODN	Object Detection Network.
RN	Relative Navigation.
SSD	Single Shot Detector.
VO	Visual Odometry.

1. Introduction

Spacecraft landing on celestial bodies, being these planets, asteroids, or comets, represents the next frontier of space exploration and has gained renewed interest in recent years, with several missions being developed by agencies and private companies all over the world. Nevertheless, it represents a very challenging phase for the Guidance, Navigation, and Control system. In fact, the delay in telecommunications between the ground segment and the spacecraft landing on a distant celestial body inhibits the feasibility of real-time controlled operations. Moreover, during landing, a change of the target landing site may be needed, for example due to a failure of one of the spacecraft subsystems or because of the detection of hazards near the originally selected landing site. For some missions, the target landing site or even the surface characteristics of the celestial body may be unknown at the

mission design phase. In all these cases, an autonomous navigation system able to manage the entire landing phase in real-time becomes an essential tool, if not the only way to ensure operational safety and provide the necessary flexibility for the mission. Vision-based navigation accomplished with a monocular camera represents a possible approach, beneficial in terms of cost and system complexity containment compared to active ranging techniques (i.e. LIDAR and RADAR). Additionally, cameras are easy to accommodate on the lander, due to their low power, mass and volume. According to [1], two different navigation functions can be identified: Relative Navigation (RN) and Absolute Navigation (AN).

The former is primarily used when the spacecraft is flying above unknown terrain, meaning that only features with unknown coordinates can be used. Depending on the application, only the relative position and orientation with respect to the previous frame or with respect to a map of the landing site, constructed during landing using images acquired by the camera sensor, can be computed [1]. An interesting preliminary design of a Relative Navigation algorithm is presented in [2]. The algorithm is designed to work without prior knowledge of the landing environment, whether this is a planet or a smaller celestial body. The proposed formulation adopts a Visual Odometry (VO) approach, using point features as navigation landmarks. Such features are detected from images using the ORB detector and tracked in consecutive frames. Subsequently, the relative pose is derived directly using the epipolar constraint. Pose refinement using Bundle Adjustment together with the development of a mapping thread are suggested, but not implemented. Tests with synthetic Moon images show robustness under varying illumination and camera pointing conditions, but the accuracy remains insufficient for autonomous navigation. Despite this, the approach offers a valid preliminary architecture that relies solely on camera measurements.

AN, instead, is used to establish position and orientation of the spacecraft with respect to a planetocentric reference frame. Features with known location are exploited for this application, requiring at least a partial knowledge of the landing environment. Specifically, feature locations are stored in a database, constructed from orbital reconnaissance acquired before landing. This highlights how employed landmarks shall be visible from orbit. The Absolute Navigation process involves detecting the selected navigation landmarks in images and then matching them with the constructed database. This information is then used to retrieve the absolute camera pose. Craters are proposed as a viable landmark for Absolute Navigation for different celestial bodies, including the Moon, due to their abundance. Based on this, several crater-based navigation algorithms have been developed over the years [3–6]. [3] and [4] propose a navigation algorithm for a generic interplanetary body which relies on a very efficient Crater Detection and Matching Algorithm (CDMA) based on standard image processing techniques. Craters detection consists of five steps: Edge detection, Rim Edge Grouping, Ellipse Fitting, Ellipse Refinement, Crater Confidence Evaluation. Subsequently, the detected craters are matched with a database exploiting conic invariants. Despite subpixel crater localization accuracy, the algorithm lacks of adaptability and robustness to different imaging conditions, limiting heavily its operability in a real-landing scenario. To overcome such limitations of traditional crater detection methods, Machine Learning-based alternatives have been developed in the last years. One of the most common solutions is to rely on Convolutional Neural Network (CNN) for image segmentation. [5] builds its algorithm based on this idea, combining a U-Net with standard image processing techniques for the edge detection task. The extracted edges are fitted using low eccentricity ellipses and then paired with craters in the database. Tests with real Moon images from the Lunar Reconnaissance Orbiter Camera show that the alternative proposed is robust to noise and brightness variations, with good detection and matching performances. The main drawback is represented by the computational time (about 4 s per image), too high for a real-time application [7]. [6] tackles this issue, renouncing to extract the crater edges but detecting only the bounding boxes around craters. For this scope, an Object Detection Network (ODN) is used, and the extracted landmarks are matched with the database using a traditional 1-Nearest Neighbor routine. The Implicit Extended Kalman Filter (IEKF) developed in [8] is employed for pose estimation, fusing visual-based measurements with altimeter ones. The developed solution provides good detection and matching performances, good localization accuracy and low computational time on flight-proven hardware. However, its application is specific to Nadir-pointing cameras. This represents a significant limitation for a landing scenario, where multiple and different pointing conditions are experienced throughout all phases.

Nevertheless, independently of the performance of specific implementations, both Absolute and Relative Navigation have some inherent limitations. For Absolute Navigation, as discussed previously, database landmarks are the ones detectable from orbit imagery. Consequently, the number of observable known landmarks decreases throughout descent, preventing state estimation during crucial final phases (altitudes below 15-20 km). Instead, for Relative Navigation, due to error accumulation, after some time only a poor estimate of the state is available. Combining information from both Relative and Absolute

Navigation could solve these problems, with the latter correcting the accumulated drift and the former extending the robustness and flexibility of the navigation algorithm. This solution is called Integrated Navigation [9].

There has been limited research on the effectiveness of this alternative, especially for the Moon environment. Although the results presented in [9] and [10] suggest that Integrated Navigation could represent a viable option for planetary landing, additional analyses are required to better assess the performance of this approach and to understand if it can effectively address the issues of single navigation, especially in the context of a lunar landing. The Moon is selected as a development scenario due to its high importance for space exploration in future years. Specifically, this paper focuses on the design of the image processing block for both Absolute and Relative Navigation threads. The selection arises from the fact that existing image processing algorithms fail to meet the accuracy and time requirements for a real-time landing application, or lack of robustness to the wide spectrum of conditions that could be encountered during a mission. Moreover, a specific pipeline for the Integrated Navigation approach needs to be developed to efficiently perform fusion between known and unknown landmark information.

The primary objective of this work is to develop image processing algorithms able to provide high-quality information under various environmental conditions while maintaining real-time performance. Known and unknown landmarks suitable for the discussed application need to be identified, along with developing an efficient strategy for their detection in images. Additionally, the performance of both algorithms shall be evaluated by extensively testing them using images simulating a representative landing scenario. In the second place, using the results of the design, the aim is to investigate the potential in the exploitation of fusion between information from both known and unknown landmarks to enhance Optical Terrain Relative Navigation on the Moon.

The following sections are structured as follows. In Section 2, the reference Moon landing scenario is presented, detailing also the camera model used, and the navigation threads frequency requirements. In Section 3 and Section 4, the proposed architecture for both the Relative and Absolute Navigation image processing blocks is detailed, from the adopted navigation landmarks to their detection and matching process. Simulations results and achieved performance are discussed in Section 5, while conclusions are drawn in Section 6, with highlights for planned future steps.

2. Moon landing: scenario definition

The mission scenario considered is the same as [6], consisting of a spacecraft descent from an altitude of 100 km down to 3 km, targeting the Lunar South Pole area. The reference trajectory is generated using optimal guidance considering the same target location and thrust constraints of [6]. For this preliminary analysis, only a portion of this trajectory is considered, with the camera still undergoing roto-translational motion with respect to the surface. Despite focusing on just part of the trajectory, the employed scenario is representative of a complete landing. The navigation algorithm will, in fact, face critical illumination and shadowing conditions due to the selected landing location. During the coasting phase, the lander travels half of the transfer orbit, covering 180° in true anomaly. This results in a highly variable ground illumination, with the Sun just above the horizon in polar regions to an elevation of approximately 90° near the Lunar Equator. The selected trajectory also covers different terrain textures to assess the performance of the developed algorithm when applied to low-texture environments.

2.1 Camera model and processing time requirements

A pinhole camera with 40° Field Of View (FOV) and a 1024×1024 pixels resolution is assumed as the main navigation sensor.

It is known that the entire navigation system is effective only with a minimum frequency of the trajectory update of 0.2 Hz (5 s) [6]. Taking some margin, a minimum required frequency of 1 Hz is assumed for the RN block. On the other hand, considering the AN block working slower than the RN one, it is required to process an image in no more than 5 s. Lower computational times would be beneficial.

3. Relative Navigation image processing architecture

For the Relative Navigation thread, a VO approach is adopted due to the nature of a landing trajectory, which, regardless of the celestial body, is open. Consequently, the same scene is not revisited more than once, preventing the exploitation of loop closure to reduce drift.

In order to estimate camera position and orientation, key information must be extracted from images. For this scope, feature-based methods, which use image features such as corners and edges, are preferred over appearance-based (direct) methods due to their superior speed, accuracy [2], and robustness to noise [11] and their good invariance to viewpoint and illumination changes [12]. This makes them particularly suitable for lunar landing and space applications in general. Point features are chosen as navigation landmarks due to their uniqueness, high repeatability and availability anywhere on the Moon. However, the performance of such methods for terrains with

low-texture needs to be assessed.

Feature detection is the first processing step of the images, providing the locations of the identified features. The goal is to consistently detect the same features across images, enabling the use of multiple view geometry to deduce information about the camera position and orientation. In real-case scenarios such as a landing descent, features experience scale variations due to decreasing altitude and changes in appearance due to different illumination conditions and viewpoints. Consequently, invariance to these factors is essential for a detector to work properly. Considering all these aspects, the detector selection is performed considering that it shall detect repeatable features, and exhibit both scale and rotation invariance and resistance to illumination changes, while also being computationally efficient for real-time applications. Different alternatives are considered and compared: FAST, SIFT, SURF, ORB, BRISK [13–18]. Among these, ORB outperforms all the other detectors across all considered metrics.

However, simulations show an insufficient number of features detected in low-texture scenes (i.e. images from 6 to 8 in Fig. 1). This limitation is inherent in the selected ORB implementation, specifically the MATLAB built-in one. In particular, the oFAST threshold, which is the minimum contrast of a point with respect to its surroundings to be classified as a corner, cannot be tuned, limiting the algorithm flexibility and operability.

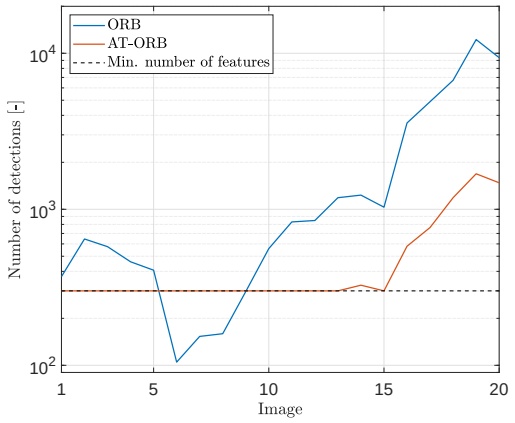


Fig. 1. Mean number of ORB and AT-ORB detections for representative frames of the landing trajectories described in Section 5.1 (mean over all trajectories).

To address this issue and enhance algorithm robustness, a customized detector is developed based on the original ORB implementation. Specifically, an adaptive oFAST threshold is employed instead of a fixed one. Following the approach in [19], the threshold th is computed based on a measure of the maximum contrast in the input image, as shown in Eq. 1:

$$th = \alpha \Delta \bar{I}_{max} \tag{1}$$

where $\alpha \in (0, 1)$ is the scale factor and $\Delta \bar{I}_{max}$ is the mean maximum contrast over the image. This is computed by dividing the image into cells and averaging the maximum contrast values from each cell, as shown in Eq. 2:

$$\Delta \bar{I}_{max} = \sum_{i=1}^{N_{cell}} \frac{I_{max,i} - I_{min,i}}{N_{cell}} \tag{2}$$

where $I_{max,i}$ and $I_{min,i}$ are the maximum and the minimum intensity over the i-th cell, respectively. In this work, 64 cells are employed. If the number of detected features is lower than the prescribed one, the threshold is halved iteratively by the algorithm until enough features are detected. An α initial value of 0.35 is considered. The developed implementation of ORB is called Adaptive Threshold ORB (AT-ORB).
Using the presented formulation, the detector is able to efficiently cope with different environmental conditions, as shown in Fig. 1. In scenes characterized by high-frequency terrain textures (i.e. images from 15 to 20), resulting in a high number of potential corners due to higher contrast, a higher threshold is applied to limit the number of detections. Conversely, in low-contrast images, a lower threshold is used (i.e. need a lower threshold to detect the same number of features). This adaptive approach ensures consistent performance across varying conditions.

The number of detections is limited to 300, which represents the upper limit for state-of-the-art CPU for space systems for the studied application [2]. Once keypoints have been extracted from an image, as already discussed, the aim is to find them in consecutive images to establish correspondences between features seen from different views. Feature tracking using the Pyramidal Lukas-Kanade algorithm [20] is selected for this purpose, showing higher performance compared to feature matching, providing more correct features correspondences per frame at a lower computational cost. However, extracting only the best 300 features, these will concentrate in corner-rich zones only, leaving areas of the image uncovered, as shown in Fig. 2. This can be considered as a loss of information for the motion estimation.

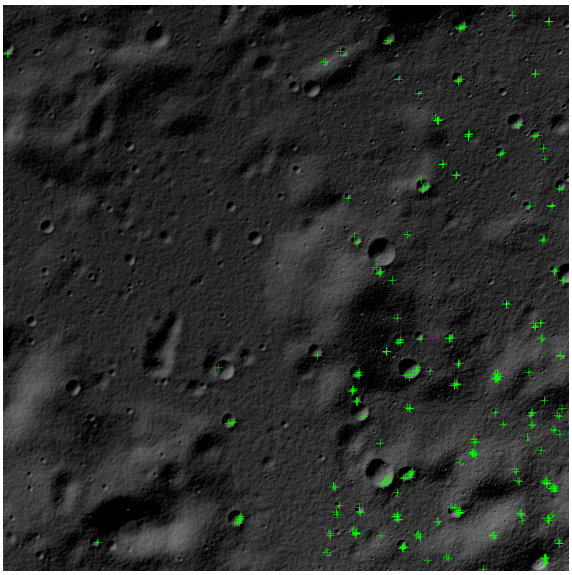


Fig. 2. Features detected by AT-ORB.

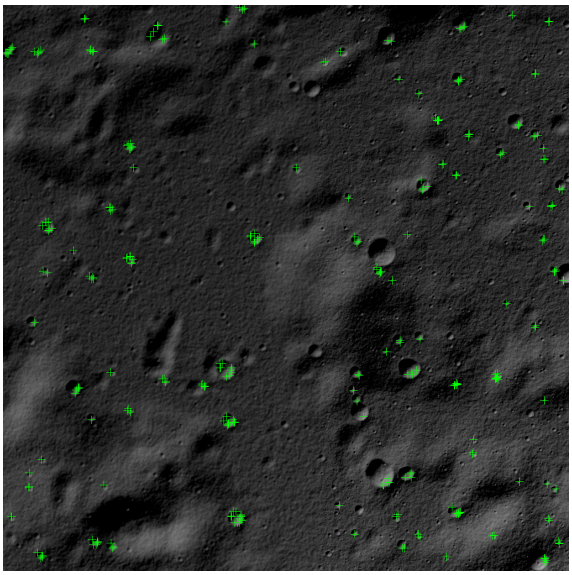


Fig. 3. Features detected by AT-ORB with 16 subwindows.

To solve this issue, the image is split into subwindows. Features are detected in each subwindow using the developed AT-ORB, and then the results are merged. Using subwindows gives better uniformity of the feature distribution, positively impacting on the whole navigation algorithm performance, leading to a higher quality motion estimate. 4, 16, 64 and 256 subwindows configurations are compared considering spatial homogeneity of the detected features, achieved features repeatability and computational time. A sliding-window within a cycle is used to move through subwindows. From tests, the 16 subwindow configuration is selected as it provides the best balance between computational time and feature homogeneity, while also ensuring good feature repeatability. Results of the Adaptive Threshold ORB detector in the 16 subwindows configuration are shown in Fig. 3.

The output of the Relative Navigation image processing thread is represented by 2D-2D correspondences between consecutive images. In fact, the algorithm is intended for use in conjunction with the IEKF developed in [8]. The filter will also be responsible for the fusion with Absolute Navigation thread measurements. Essential matrix is selected as model for the outlier rejection. Specifically, the 5-Point Algorithm [21] with MSAC [22] is used to retrieve only inliers correspondences, achieving good performance independently of the scene structure and with a low computational time. When the number of tracked or inliers features becomes lower than a threshold (50), detection is performed on the previous frame and found features are tracked at current time to achieve correspondences.

4. Absolute Navigation image processing architecture

Before delving into the design of the Absolute Navigation image processing thread, suitable navigation landmarks shall be identified. Considering that the objective is to develop a navigation algorithm capable of operating anywhere on the Moon, a landmark is suitable only if it is present everywhere. Craters represent viable landmarks for Absolute Navigation during lunar landing due to their abundance on the lunar surface [6]. However, only a low number of craters is present in the Lunar Maria, which are instead characterized by a high concentration of rilles and wrinkled ridges, as shown in Fig. 4 and Fig. 5. The utilization of such features, referred to as line features, as navigation landmarks has been demonstrated for Mars landing by [9]. The idea is to exploit the same strategy for navigation on Lunar Maria, given their similarities to the Martian surface. Therefore, a combined approach using craters along with rilles and ridges as navigation landmarks is proposed for achieving global coverage of the lunar surface, as shown in Fig. 6.

4.1 Proposed Absolute Navigation architecture

Both craters and line features shall be detected and then matched with a database. To reduce processing time, it is proposed to initiate line features detection and matching only when the number of detected or matched craters falls below a fixed threshold, for which poor Absolute Navigation information are available.

This paper focuses only on the implementation of the crater-based Absolute Navigation algorithm, as existing

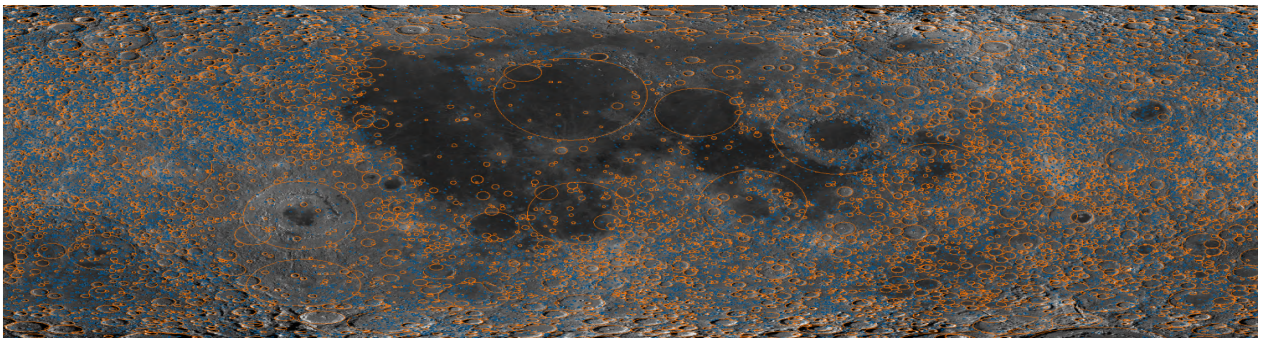


Fig. 4. Craters with diameter >5 km. In blue craters from 5 to 20 km; in orange craters >20 km. As highlighted, few craters are present in Lunar Maria. Image taken from [23].

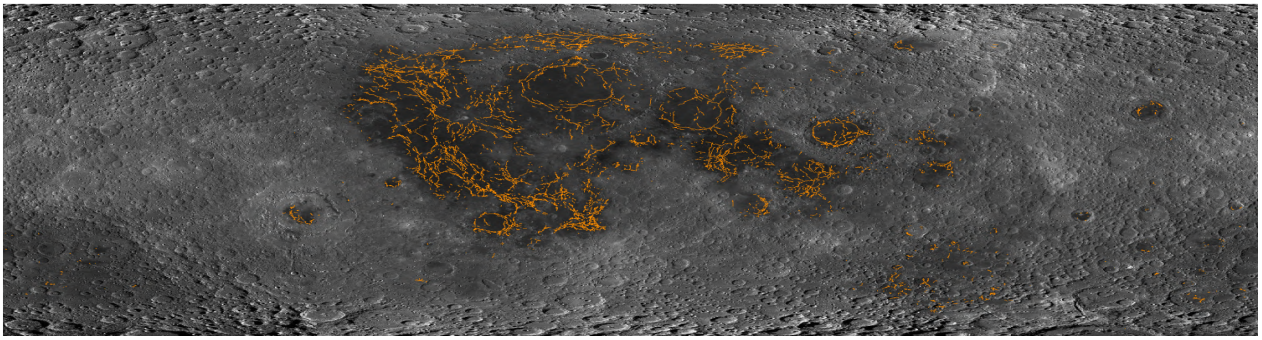


Fig. 5. Line features distribution. In the figure, only wrinkled ridges are represented. Image taken from [23].

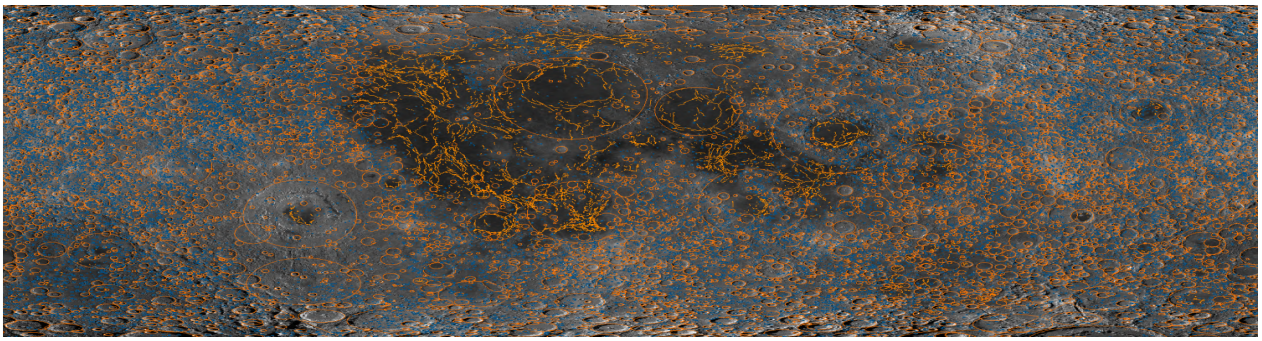


Fig. 6. Craters with diameter >5 km and wrinkled ridges distribution. Global coverage of the Moon surface is achieved. Image taken from [23].

crater-based algorithms, although capable of achieving sufficiently accurate navigation performance, are developed exclusively for Nadir-pointing cameras and either are computationally intensive (e.g. [5]) or lack of robustness to environmental conditions (e.g. [3]), making them unsuitable for real-time operations. It shall also be considered that the crater-based navigation remains the baseline for the Absolute Navigation, being craters available on the majority of the lunar surface. Line features navigation, although fundamental, serves as a backup to increase the robustness of the whole navigation algorithm.

The objective is thus to design a Crater Detection and Matching Algorithm (CDMA) able to achieve a level of navigation accuracy comparable to the best state-of-the-art algorithms while ensuring sufficiently low computational time and robustness to various scenarios, including different environmental conditions and viewing angles of the lunar surface.

4.2 Crater detection

Crater detection is the process of identifying and extracting the rims of craters in an image. For this purpose, ML-based methods are considered due to their high robustness compared to traditional techniques, still being able to achieve real-time performance [6]. A generic ML-based crater detection algorithm can be subdivided into two subsequent steps, namely *crater localization* and *crater fitting*. Crater localization involves identifying the crater rims in the image using a CNN dependent on the specific approach. However, the rim is usually only partially recovered, or only its bounding box is extracted. To obtain the complete curve of these rims, whose information will be used for the matching process, crater fitting is subsequently performed.

4.2.1 CNN architecture selection

Different CNN-based detector architectures exist depending on how the crater problem is approached. The majority of previous works, such as LunaNet [7] and DeepMoon [24], address the crater localization as a segmentation problem. Indeed, detecting crater rims is inherently a semantic segmentation task, as it involves extracting fine-grained information (i.e. need to extract the rim pixels) [5]. However, semantic segmentation, despite its high accuracy, has a high computational cost, which limits the algorithm capability for real-time performance, as pointed out in [6]. For this reason, the approach proposed by [6] is followed in this work, utilizing an Object Detection Network to locate craters. Specifically, a YOLOv7 network [25] is used to identify the bounding boxes surrounding each crater, being one of the fastest and most accurate state-of-the-art ODN.

4.2.2 Training dataset definition

A Digital Elevation Model (DEM) of the Moon has been exploited to generate simulated images taken by the lander navigation camera in order to create the training dataset. Specifically, two subdatasets are created. The first one, called "*spherical-moon*", is based on SLDEM2015, a real Moon DEM derived by combining the DEMs from the LRO and SELENE missions [26]. The constructed subdataset, although it represents a high-fidelity terrain scenario of the Moon surface, is suitable only for simulating images taken from high altitude due to the limited resolution of the DEM. For this reason, a second fully synthetic subdataset, called "*flat-moon*", is constructed upon real Moon features distribution. A flat DEM is created, perturbed with fractal noise and enriched with the other relevant terrain features, i.e. craters. The lunar impact crater size and distribution have been extracted from [27]. For both cases, the portion of the Moon used for generating the images is modelled in PANGU, a high-fidelity rendering software meant for space applications and realistic rendering of natural celestial bodies [28]. The crater distribution characteristics for the synthetic generation performed in PANGU are reported in Table 1.

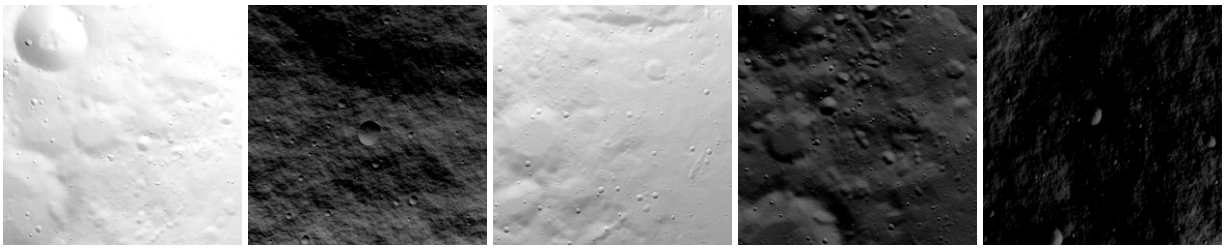
Table 1. Environmental variables and their range of variation for the dataset generation.

Variable	Range
Synthetic crater frequency	1.8e6 - 3e6
Synthetic craters dimension	6 - 500 m
Altitude	3 - 100 km
Attitude pitch (wrt vertical)	0° - 20°
Sun illumination angle - Elevation	0° - 90°
Sun illumination angle - Azimuth	0° - 360°

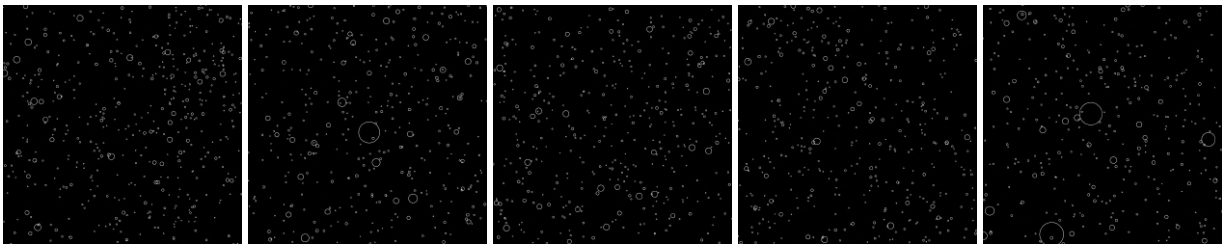
To create a rich and representative dataset, the environmental variables in Table 1 are randomly varied within the reported ranges. In this way, the dataset can cover the wide variety of environmental conditions that are expected in the operational scenario. These different conditions can also be experienced during the same landing trajectory, since the lander travels half of the transfer orbit during the coasting phase. In particular, the inclination of the Sun over the terrain may change significantly, from the Sun slightly above the horizon in polar regions to potentially straight illumination with 90° of Sun elevation close to the Lunar Equator. A range between 0° to 360° is considered for the Sun Azimuth angle, although this is applicable only for high latitudes close to the Poles.

For the dataset, 1024×1024 grayscale images are generated according to the camera model described in Section 2.1. Specifically, 5000 images have been created for the "*spherical-moon*" subdataset and 10000 for the "*flat-moon*" one. For each image of both subdatasets, the list of visible craters with the corresponding centre coordinates and radii is available. The origin of the reference system is placed in the upper-left corner of the image.

The actual training dataset consists of all the 5000 "*spherical-moon*" images and only 5000 "*flat-moon*" selected images, ensuring equal representation of conditions from the two subdatasets. The combination of high-fidelity simulated images from "*spherical-moon*" and synthetically generated images from "*flat-moon*", together with the presence of various ground illumination conditions and textures, provides a representative and rich dataset for training the object detection model successfully. Some examples of images stored in the dataset are reported in Fig. 7a, with the ground truth craters present in the images highlighted Fig. 7b.



(a) Images.



(b) Ground truth craters.

Fig. 7. Training dataset examples.

Given the selected 10000 images, an additional step is needed to create the actual training dataset. First, the images are downsampled to 640×640 px resolution and then converted to a 3-channel image, resulting in a final size of $640 \times 640 \times 3$ px, which matches the original input size of the YOLOv7 network. The conversion from a single channel to RGB is computationally negligible, as the information from the original channel is simply replicated across the three channels. The downscaling is chosen for two main reasons: firstly, it maintains the input size consistent with the original requirements of the YOLOv7 network; secondly, increasing the input size would impact the inference time, which is critical for the algorithm implementation on real hardware [6]. For each image, also the available annotations (i.e. centre coordinates and radii) shall be converted into the correct YOLOv7 format, consisting of the bounding boxes attributes: class, centre coordinates, width, and height. A unique detection class (i.e. "*crater*") is considered for this work. The conversion from the original annotations to the requested ones is quite straightforward. Considering a square bounding box bordering each crater, both width and height are computed as the double of the radius, while the centre coordinates coincide with those of the crater. All box quantities are then normalized by the image size, constraining their interval between zero and one.

After all these steps, the whole dataset is set up for YOLOv7 training, and it is composed as follows:

- The set of images of the lunar terrain with resolution $640 \times 640 \times 3$ px;
- The list of the visible craters in each image, with annotations containing the normalized bounding boxes centre coordinates, width, and height for the 640×640 resolution. For each crater, annotations are in form [0 x_c y_c w h].

4.2.3 YOLOv7 training

From the official YOLOv7 repository [29], a pretrained model on the MS COCO dataset is available. Utilizing this pretrained model enables the application of transfer learning to accelerate the training process, achieving high performance with a relatively small dataset like the one presented in the previous section. The adopted loss function is the one from the original implementation contained in [29], and it is composed of three contributions:

$$L(cl,\ pred,\ gt) = \frac{1}{N}[\lambda_{box} L_{box}(pred, gt) + \lambda_{cls} L_{cls}(cl) + \lambda_{obj} L_{obj}(cl)] \quad (3)$$

Here, L_{box} represents the regression loss for bounding box coordinates, expressed as the Complete IoU (CIoU) loss [30] between a predicted bounding box $pred$ and the ground truth one gt. L_{cls} is instead the classification loss, which represents the confidence level that the image contained in the predicted box corresponds to a particular class cl, while L_{obj} is the objectness loss, representing the confidence that an object is present within the predicted bounding box $pred$.

The training is performed using the SGD optimizer

with an initial learning rate of 0.01 for 300 epochs, using a batch size of 32. The optimal training configuration is found performing faster preliminary training runs. To enhance network robustness, data augmentation techniques are applied, significantly altering the appearance of craters within the dataset. The dataset is split into training, validation, and test sets, in 70-10-20% percentages, respectively.

4.2.4 Crater localization post-processing

The output of the YOLOv7 network is a matrix of dimensions $n \times 6$ reporting the list of craters located in the image. Each row corresponds to a single detected object and is a 6-element vector: $[x_{min}\ y_{min}\ x_{max}\ y_{max}\ cl\ \alpha]$. The first four elements are the coordinates of the bounding box enclosing the crater, expressed in pixels with values ranging from zero to the input image size. The attribute cl represents instead the object class associated to the bounding box, which is 0 for all detections, as the network is trained to detect exclusively the "*crater*" class. Last, the index $\alpha \in [0, 1]$ is a score representing the network confidence in the crater identification: a low score translates to little confidence that the output coordinates correspond to the bounding box of a crater. Only craters detected with high confidence $\alpha \geq \overline{\alpha}$ are considered for the subsequent crater fitting step. Moreover, the Intersection over Union threshold $\overline{\text{IoU}}$ is used in Non-Maximum Suppression (NMS) to remove multiple boxes that surround the same object, selecting the one with the highest confidence. Both $\overline{\alpha}$ and $\overline{\text{IoU}}$ are adjustable parameters of the navigation system: in the remainder of the work, $\overline{\alpha} = 0.6$ and $\overline{\text{IoU}} = 0.65$ are assumed. The number of detections is limited to 70 for computational purposes.

4.2.5 Crater fitting

Given the detected bounding boxes, crater fitting is needed to approximate crater rims and retrieve crater attributes. In the literature, detected craters are usually fitted using circles (or ellipses with low eccentricity). While this approach is suitable for Nadir-pointing cameras, it imposes a significant limitation on the algorithm's operational flexibility. In fact, although the majority of craters are circular when viewed from above, they appear as ellipses, generally oblique and often with high eccentricity, due to viewing angle and camera relative position. For this reason, in the detector implemented in this work, craters are fitted with ellipses, without limitations on eccentricity, thereby extending algorithm operability to any camera pointing condition and filling a gap in the existing literature.

A method to fit an oblique ellipse into each bounding box is thus developed, generalizing the procedure presented in [6]. The proposed approach assumes that all craters appear as circles to a Nadir-pointing camera and transform into ellipses, all inclined at the same angle, considering a change in viewing angle. First, the crater inclination angle, equal for all craters under the previous assumption, is estimated using standard image processing techniques applied to a limited set of detected bounding boxes. The obtained information is then used to solve, for each bounding box, the non-linear system of equations that defines the inscribed ellipse problem. As output, the parameters of the ellipse approximating the crater rim are obtained for each bounding box. Results of the proposed crater detection strategy are shown in Fig. 8.

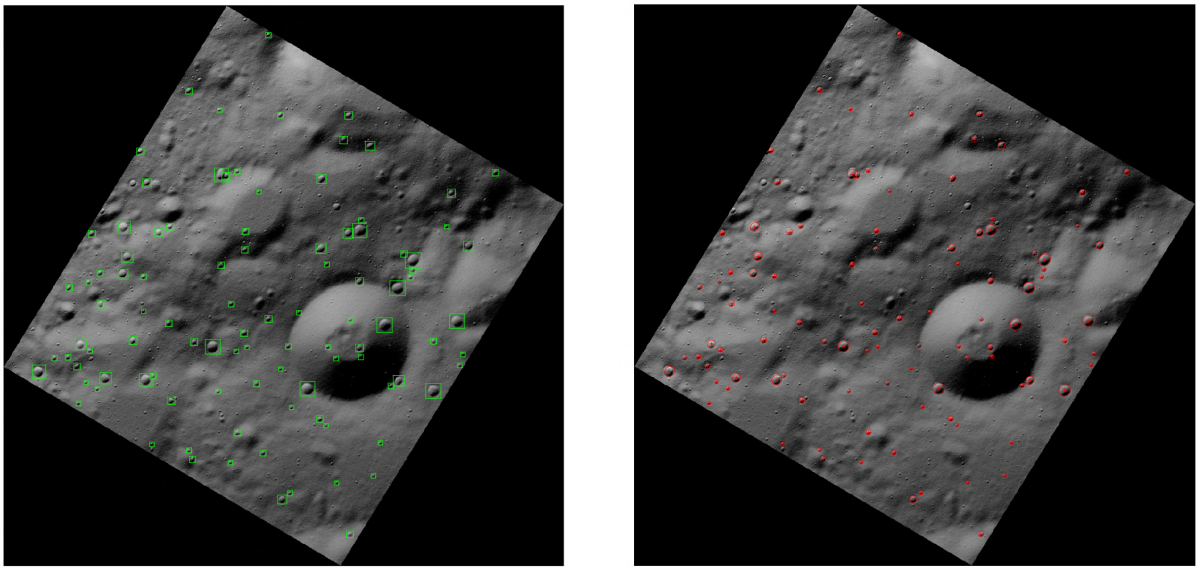
4.3 Crater matching

The next task in the crater-based Absolute Navigation pipeline involves matching the detected craters, expressed in image frame coordinates, to real database craters, which are instead expressed in a Moon-fixed frame. In this way, the absolute location of identified landmarks can be determined and later used to retrieve useful information about the camera pose. The terms *catalog* and *database* are used interchangeably in the literature and also here.

Generally speaking, a match can be found by comparing certain parameters of the detected crater to those of the craters in the database. However, the global crater catalog considered for this work [31] is massive, containing more than 2 million elements. Without any additional constraints, the search for a match would be exceedingly time-consuming. To restrict the search space, the a-priori pose estimate coming from the navigation filter is utilised following the procedure employed in [6]. Specifically, the available estimated pose is considered as the true one, and the FOV of the camera is projected onto the Moon surface using spherical projection. The output of the projection is a set of corners, representing the boundaries of the projected FOV, with each corner expressed in latitude and longitude coordinates. Only the database craters belonging to this box are considered for the matching procedure.

The extracted database craters coordinates are then projected into the image frame using the pose knowledge coming from the navigation filter. Although this approach results in a loss of computational efficiency, as the number of extracted database craters will still be higher than that of the detected ones, it enhances the descriptiveness of the crater parameters due to their elliptical appearance.

Next, craters are matched by proximity and similarity using the 1 Nearest-Neighbor Search (1NNS) algorithm, comparing ellipse parameters of both detected and projected catalog craters. To enhance search efficiency and facilitate the matching procedure, crater catalog data are organised in a KD-tree [6]. The Euclidean distance is employed as dissimilarity metric, with a multiplicative cor-



(a) Crater localization.

(b) Crater fitting.

Fig. 8. Example of crater localization and fitting. Bounding boxes are converted into ellipses approximating crater rims.

rection to consider different order of magnitude among ellipse attributes. However, unsatisfactory performance are exhibited by this implementation if a very poor pose estimate is available. An iterative outlier rejection and rematch strategy is thus designed to enhance matching performance and robustness, reducing uncertainty on the available pose knowledge. This strategy iteratively refines pose estimates and reprojects catalog craters using the refined poses to improve match accuracy, as suggested in [32]. In fact, being the refined pose estimate more accurate than the a-priori one, additional correct matches are expected to be found. This iterative procedure, referred to as the Matching Loop, is stopped when a sufficient number of correct matches is achieved or the maximum number of iterations is reached.

5. Algorithm Testing and Performance

Simulations, unless differently stated, are performed on MATLAB on an Intel® Core™ i7-7820HQ CPU processor.

5.1 Relative Navigation image processing

The performance of the algorithm are assessed using a dataset composed of synthetic images of five lunar landing trajectories. These trajectories are generated starting from the reference one presented in Section 2 using the high-fidelity dynamics simulator implemented in [6]. This block takes the presented nominal guidance profile and simulates the overall landing manoeuvre in a high-fidelity scenario. To account the Moon gravitational pull in the translational dynamics, the LP165P spherical harmonics model up to the 165th order [33] is adopted. Disturbances in both direction and magnitude of thrust are included. Specifically, the thrust vector is rotated by a random angle represented by a normal distribution with zero mean and standard deviation $\sigma = 1^\circ$, while the thrust magnitude is perturbed by a Gaussian noise with standard deviation 23 N (1% of the assumed throttleable thrust). The navigation camera is assumed to maintain a nominal Nadir pointing, with a Gaussian noise with standard deviation $\sigma = 1^\circ$ added to the three Euler Angles to represent attitude determination errors [6]. For each trajectory, setting the camera sampling frequency equal to 1 Hz, a sequence of 1000 consecutive images is generated using the same DEM employed for the "*spherical-moon*" subdataset in Section 4.2.2. To construct a dataset representative of the possible environmental conditions that the algorithm may face, different Sun illumination directions are considered for each trajectory. Each sequence simulates what the lander camera observes during the corresponding landing manoeuvre. The testing dataset, referred to as *trajectories dataset*, is formed by the union of these five sequences. The such composed dataset encompasses a variety of critical scenarios, not only from illumination and terrain conditions but also due to motion. The objective is to stress the navigation algorithm to understand not only its performance, but also its robustness to all conditions that can be encountered during a landing. This is the same dataset

used for the thread design presented in Section 3.

The whole Relative Navigation block is tested using sequences of images derived from the constructed trajectories dataset. The developed AT-ORB detector in the 16 subwindows configuration successfully detects the prescribed number of features (300) in all images, with an average computational time of 42.5 ms. Moreover, the Pyramidal Lukas-Kanade algorithm effectively tracks 91.8% of the detections. Of these tracked features, the 93.3% are correct correspondences, efficiently retrieved by the proposed outlier rejection strategy. Specifically, the 5-Point Algorithm with MSAC takes 68.7 ms on average. In the worst-case scenario, where redetection is triggered due to an insufficient number of inliers (i.e. perform detection once and both tracking and outlier rejection twice), the whole Relative Navigation thread takes about 214.1 ms on average to process an image. This corresponds to a frequency well above the required minimum of 1 Hz, making the algorithm suitable for a real-time application.

5.2 Crater-based Absolute Navigation image processing

The performance and robustness of the algorithm are assessed using three different synthetic datasets:

1. the trajectories dataset described in Section 5.1;
2. the test set of the YOLOv7 training dataset;
3. the *matching dataset*, built by applying an affine transformation to 500 training dataset images to simulate different perspective views. For each image, available crater annotations are used as a prefiltered catalog, while the affine transform also serves as ground truth pose.

Crater localization is tested on both the test set and the trajectories dataset, with the latter included to evaluate the detector in an entirely different environment from the one used for training. Both matching and fitting, on the other hand, are tested on the third dataset, which incorporates all the aspects that add complexity to the fitting process. For the matching, noise is introduced into the affine transform parameters to assess the algorithm robustness under various levels of accuracy in the available pose knowledge. Two noise levels, high and low, are considered to test the algorithm robustness and performance under different conditions. The perturbed affine transform is then used as the a-priori estimate of the camera pose provided by the filter.

5.2.1 Crater localization

Performance metrics of the crater localization evaluated on the test set are reported in Table 2.

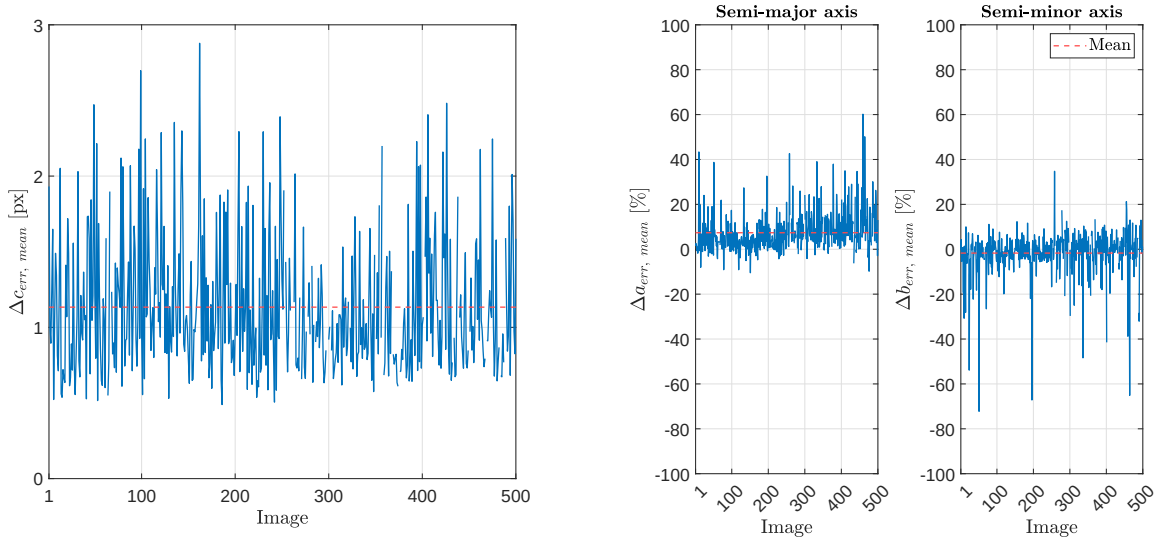
Table 2. Crater localization performance metrics on the test set.

Metric	Value
Precision	0.994
Recall	0.493
F_1 score	0.659
$mAP_{0.5}$	0.498
$mAP_{0.5:0.95}$	0.297
Mean IoU	0.772

On average, the network is able to retrieve only approximately 50% of annotated craters. However, all detections are practically correct, with only a minimal percentage (less than 1%) of false positives. Compared to other state-of-the-art CNN-based crater detectors, the implemented network achieves slightly lower performance. Specifically, this is due to the developed network's recall, which is approximately 40% points lower than that of both the SSD with MobileNetV2 used in [6] and DeepMoon [24], resulting in a lower F_1 score. Despite this, higher detection capacity of the network is indeed observed. In fact, looking at the state-of-the-art alternatives, the [6] SSD is able to detect nearly 50 craters per image, while LunaNet [7] only 20. The trained YOLO network, instead, is able to detect almost 200 craters on the test set and 70 on the trajectories dataset. The lower detection count for the latter subdataset is not linked to any issues in the detection framework or lack of generalization, but rather to a lower number of craters effectively present in the image, as confirmed by visual inspection. Additionally, the YOLO network achieves a higher mean IoU compared to [6]. Being the mean IoU linked to localization accuracy, the developed detector is expected to achieve a lower crater localization error. Inference time, instead, is suitable for a real-time application, taking 16.3 ms on the test set and only 12.7 ms on the trajectories one [‡].

Examining the images relative to the test set, such as Fig. 8a, one issue can be noted: large craters in the images are not detected. This limitation is inherently linked to the YOLOv7 network, which struggles to detect an object if it appears in different dimensions or aspect than the training data. The undetected craters were, in fact, non annotated and, most importantly, show a different appearance (e.g. fuzzier rim, different shading) with respect to the smaller annotated ones. Adding annotations for large craters may solve this issue. Another potential solution is to use anchor-free detectors, which do not rely on predeter-

[‡] Inference time is evaluated in Python on a 13th Gen Intel® Core™ i7-13800H CPU processor.



(a) Centre location.

(b) Semi-major and semi-minor axes.

Fig. 9. Crater fitting errors over images.

mined anchor boxes. Consequently, they are expected to provide higher generalization capability and provide more accurate bounding boxes in size and aspect ratio. It is relevant to note that the presented issue is present only in the test set images, while the trajectories dataset includes detections of large craters. Nonetheless, this lack of generalization results in a significant problem for a real autonomous navigation application of a spacecraft using the YOLOv7 network in the considered configuration, representing a loss of robustness for the entire navigation algorithm. While this loss can be mitigated for the Moon due to the high availability of imagery data to build a sufficiently representative training dataset, it poses a relevant limitation to the network applicability to other celestial bodies for which data may not be available until arrival.

Despite this, the number of detections retrieved is more than sufficient for a navigation application. Additionally, smaller craters are detected, showing an improvement compared to other methods, both CNN-based and traditional. This capability is independent of the altitude of the images, being the size of the craters intended in pixels. The detector can thus provide a sufficient number of detections throughout the whole landing descent.

5.2.2 Crater fitting

Crater detection achieves an average localization error of approximately 1 px, with errors consistently below 3 px, as shown in Fig. 9a. This performance is linked to the high mean IoU achieved by the trained YOLOv7 model. The average errors for the semi-major and the semi-minor axes, shown in Fig. 9b, are approximately 7% and -2%, respectively, with a 3σ error band of about 30% around the mean values. The proposed implementation exhibits significantly superior localization accuracy compared to other state-of-the-art algorithms, providing the highest accuracy among developed CNN-based methods. It achieves levels of accuracy comparable to edge-based detectors, while being more robust to environmental conditions and detecting a much higher number of craters. However, the main criticality of the developed crater fitting algorithm is its computational time, equal to 512.2 ms on average but with peaks exceeding 1 s. Moreover, the procedure fails for a limited number of images (22 out of the 500 images of the matching dataset across multiple simulations, representing the 4.4% of occurrence), all depicting a dark scene with the presence of a high-frequency terrain texture. This inefficiency is primarily due to the employed implementation for the lighting source direction estimation procedure in the crater inclination estimation routine.

5.2.3 Crater matching

The Matching Loop, instead, works robustly across all dataset images, even considering high noise on the pose knowledge, correctly matching practically all detections in an average time of 61.3 ms. This efficiency is attributed to the powerful outlier rejection strategy, which reliably identifies and discards all outliers.

Overall, the entire crater-based Absolute Navigation thread processes an image in approximately 573.5 ms on average. Considering also peaks in computational time across images, the current configuration can manage one image every 2 s. Lower times may be achieved by improving the crater fitting strategy. Examples of crater detection and matching results are shown in Fig. 10 and Fig. 11.

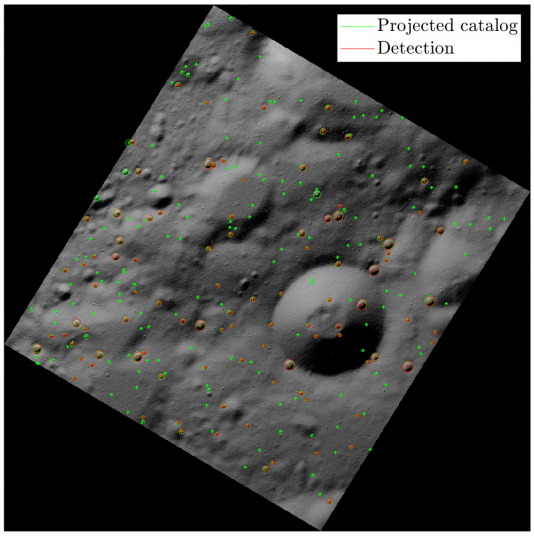


Fig. 10. Detected craters and projected database craters.

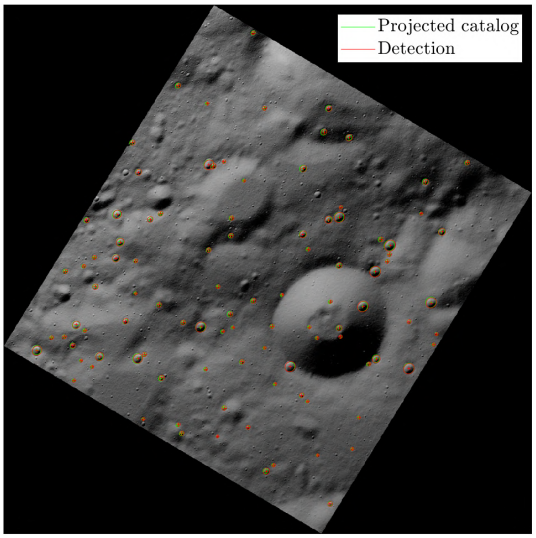


Fig. 11. Correctly matched pairs retrieved by the Matching Loop algorithm.

5.3 Integrated navigation

The impact of the developed image processing algorithms on navigation performance is assessed through comparisons with existing literature.

For the Relative Navigation, [8] is used as reference given the similar problem formulation and simulation scenario. The algorithm presented in [8] achieves a mean horizontal error of $\sim$200 m and a mean vertical error of $\sim$100 m, with 3σ bounds of approximately 1000 m and 250 m, respectively. Considering the implementation presented in this work, due to the higher number of features employed, a lower position estimate covariance is expected. In fact, increasing the number of available features reduces uncertainty on the pose [8]. Higher landing performance are thus anticipated from an uncertainty perspective. Moreover, a mean tracking error of 1.88 px is achieved, being this the error between tracked features and their true position in the image. This high feature localization accuracy, coupled with high feature homogeneity in the image, is anticipated to impact positively on the navigation algorithm, reducing the drift introduced by the RN thread and improving navigation accuracy performance compared to [8]. Low drift is fundamental, as it results in lower navigation errors when information from the Absolute Navigation are not available.

Talking about the Absolute Navigation, instead, the high localization accuracy achieved by the implemented CDMA, superior to other state-of-the-art crater detectors, together with the higher number of craters matched, is expected to have a positive impact on the overall navigation performance. Specifically, higher localization accuracy directly enhances navigation precision, while having more available features further reduces estimation covariance, as discussed for the Relative Navigation. To estimate the navigation accuracy, a comparison can be made with the results reported in [6], which uses the same filter and dataset. Based on previous considerations, the developed algorithm is anticipated to outperform this benchmark in both accuracy and covariance. Specifically, an estimation error along the trajectory lower than 200 m for both vertical and horizontal positions, with a 3σ uncertainty below 250 m, is expected with the proposed configuration. Comparing quality of camera measurements with state-of-the art attitude sensors, a pointing accuracy in the order of 1° is instead anticipated using only the camera for attitude estimation. Independently, thanks to image processing high robustness, the algorithm is expected to be suitable for a large variety of landing scenarios.

Individually, each thread of the developed navigation architecture is expected to deliver sufficiently good performance. Therefore, theoretically, integrated navigation is expected to enhance the overall performance of Optical Terrain Relative Navigation across diverse scenarios, including varying illumination conditions, camera pointing accuracy, and terrain morphology.

6. Conclusions

This paper presents a preliminary design of image processing algorithms for both Relative and Absolute Navigation to enable Integrated Optical Terrain Relative Navigation during a lunar landing.

For the RN thread, a new feature detector, called Adaptive Threshold ORB (AT-ORB), has been developed based on ORB. Thanks to its adaptive oFAST threshold, the detector is able to achieve the same number of detections independently of the scene observed, enhancing detection robustness while maintaining a low computational time. Moreover, the integration of the detector with the subwindows improves feature homogeneity across the image, leading to a more accurate pose reconstruction. The thread achieves good computational efficiency, successfully meeting the 1 Hz time requirement for the Relative Navigation.

For the Absolute Navigation, on the other hand, craters, rilles and wrinkled ridges are selected as navigation landmarks to achieve global lunar coverage. In this work, only the crater-based part of the thread is developed. The employed YOLOv7 network successfully retrieves the Moon craters in an image, delivering excellent centre localization accuracy (about 1 px error on average). However, a lack of generalization is observed for the selected ODN, failing to detect large non annotated craters, posing an issue for real autonomous navigation applications. Using anchor-free ODN may address this limitation. Once craters are detected, the Crater Fitting algorithm reconstructs the detected crater rims solely from bounding boxes, assuming an elliptical crater shape. While the semi-axes estimation is highly accurate, with mean errors below 10%, the process is slow, with time peaks exceeding 1 s. Additionally, the fitting routine struggles in the case of dark images with high-frequency terrain texture due to the too simple procedure for the illumination source direction estimation. Further modifications are thus needed to improve speed and robustness. For the matching, a robust method is developed and tested: simulations demonstrate the robustness of such a strategy even in the case of very high errors on the pose knowledge, where the 1NNS alone would fail. The algorithm successfully matches nearly all detections to the corresponding database elements, with a false match rate below 1%. The overall thread computational time is suitable for a real-time application, processing one image in less than 2 s, fulfilling the prescribed requirement.

Comparing achieved results with existing literature and assessing the impact of the proposed image processing algorithms on the navigation performance, integrated navigation is deemed as a promising strategy to enhance Optical Terrain Relative Navigation for lunar landing. However, further detailed analysis and validation are required to ensure that the proposed integrated navigation approach reliably meets all navigation requirements. This will involve rigorous testing of the navigation system under different environmental conditions to ensure both robustness and accuracy in real-world applications and to thoroughly assess the performance of the proposed implementation. Additionally, a suitable filter shall be developed to fuse information coming from the two navigation threads, accounting for processing delays, and to estimate the lander pose. For future developments, the design and integration of the line feature-based Absolute Navigation represents a crucial step to achieve the desired robustness and reliability of the whole navigation algorithm.

References

[1] A. Johnson and J. Montgomery, "Overview of Terrain Relative Navigation Approaches for Precise Lunar Landing," in *Proceedings of the 2008 IEEE Aerospace Conference*, 2008.

[2] L. Losi, "Visual Navigation for Autonomous Planetary Landing," M.S. thesis, Politecnico di Milano, 2016.

[3] Y. Cheng, A. E. Johnson, L. H. Matthies, and C. F. Olson, "Optical Landmark Detection for Spacecraft Navigation," in *Proceedings of the 13th AAS/AIAA Space Flight Mechanics Meeting*, 2003, pp. 1–19.

[4] J. He, H. Cui, and J. Feng, "Edge Information Based Crater Detection and Matching for Lunar Exploration," in *Proceedings of the International Conference on Intelligent Control and Information Processing*, 2010.

[5] L. M. Downes, T. J. Steiner, and J. P. How, "Lunar Terrain Relative Navigation Using a Convolutional Neural Network for Visual Crater Detection," in *Proceedings of the American Control Conference*, 2020.

[6] S. Silvestrini *et al.*, "Optical navigation for Lunar landing based on Convolutional Neural Network crater detector," *Aerospace Science and Technology*, vol. 123, 2022.

[7] L. M. Downes, T. J. Steiner, and J. P. How, "Deep Learning Crater Detection for Lunar Terrain Relative Navigation," in *Proceedings of the AIAA Scitech 2020 Forum*, 2020.

[8] S. Silvestrini, M. Piccinin, G. Zanotti, A. Brandonisio, P. Lunghi, and M. Lavagna, "Implicit Extended Kalman Filter for Optical Terrain Relative Navigation Using Delayed Measurements," *Aerospace*, vol. 9, 2022.

[9] W. Shaoa, L. Caoa, W. Guoa, *et al.*, "Visual Navigation Algorithm Based on Line Geomorphic Feature Matching for Mars Landing," *Acta Astronautica*, vol. 173, pp. 383–391, 2020.

[10] P. Cui, X. Gao, S. Zhu, and W. Shaod, "Landmark-based autonomous navigation for pinpoint planetary landing," *Advances in Space Research*, vol. 58, pp. 2313–2327, 2016.

[11] M. Bussolino, "Multispectral Vision-based Relative Navigation to Enhance Space Proximity Operations," M.S. thesis, Politecnico di Milano, 2022.

[12] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, 2015.

[13] E. Rosten and T. Drummond, "Fusing Points and Lines for High Performance Tracking," in *Proceedings of the 10th IEEE International Conference on Computer Vision*, 2005, pp. 1508–1515.

[14] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," *Lecture Notes in Computer Science (LNCS)*, vol. 3951, 2006.

[15] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[16] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," *Lecture Notes in Computer Science (LNCS)*, vol. 3951, pp. 404–417, 2006.

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Proceedings of the 2011 IEEE International Conference on Computer Vision*, 2011, pp. 2564–2571.

[18] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *Proceedings of the 2011 IEEE International Conference on Computer Vision*, 2011, pp. 2548–2555.

[19] G. Chu, Y. Peng, and X. Luo, "ALGD-ORB: An improved image feature extraction algorithm with adaptive threshold and local gray difference," *PLoS ONE*, vol. 18, 2023.

[20] J. Y. Bouguet, *Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm*, 1999.

[21] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 756–770, 2004.

[22] P. H. S. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry," *Computer Vision and Image Understanding*, vol. 78, pp. 138–156, 2000.

[23] A. C. T. (ACT), *Lunar/LROC QuickMap*, Last visit: 2024-05-29, 2024. [Online]. Available: `https://quickmap.lroc.asu.edu/`.

[24] A. Silburt, C. Zhu, M. Ali-Dib, K. Menou, and A. Jackson, "DeepMoon: Convolutional neural network trainer to identify moon craters," *Astrophysics Source Code Library*, 2018.

[25] C.-Y. Wang *et al.*, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," in *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[26] M. K. Barker, E. Mazarico, G. A. Neumann, M. T. Zuber, J. Haruyama, and D. E. Smith, "A new lunar digital elevation model from the Lunar Orbiter Laser Altimeter and SELENE Terrain Camera," *Icarus*, vol. 273, pp. 346–355, 2016.

[27] G. Neukum, B. König, and J. Arkani-Hamed, "A study of lunar impact crater size-distributions," *The Moon*, vol. 12, pp. 201–229, 1975.

[28] I. Martin, M. Dunstan, and M. Sanchez Gestido, "Planetary surface image generation for testing future space missions with PANGU," in *Proceedings of the 2nd RPI Space Imaging Workshop*, 2019.

[29] C.-Y. Wang (WongKinYiu), *yolov7*, Last visit: 2024-06-14, 2022. [Online]. Available: `https://github.com/WongKinYiu/yolov7`.

[30] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[31] S. J. Robbins, "A new global database of lunar impact craters >1–2 km: 1. Crater locations and sizes, comparisons with published databases, and global analysis," *Journal of Geophysical Research: Planets*, vol. 124, pp. 871–892, 2019.

[32] B. Maass *et al.*, "Crater Navigation System for Autonomous Precision Landing on the Moon," *Journal of Guidance, Control, and Dynamics*, vol. 43, pp. 1414–1431, 2020.

[33] A. Konopliv, A. Kucinskas, W. Sjogren, J. Williams, A. Binder, and L. Hood, *The Gravity Field of the Moon from the Lunar Prospector Mission*, 1998.