IAC-25,C1,IP,4,x94965

AI-ENABLED OPTICAL NAVIGATION AND ANOMALY DETECTION FOR A MOON-ORBITING SPACECRAFT

Murathan Bakır[a*] and Demet Cilden-Guler[a,b]

[a] ***Department of Astronautical Engineering, Istanbul Technical University, Sarıyer/Istanbul, 34469, Turkiye***

[b] ***Department of Mechanical and Manufacturing Engineering, University of Calgary, Calgary/Alberta, T2N 1N4, Canada***

Abstract

Autonomous navigation and attitude determination systems are essential in the cis-lunar environment, with optical navigation being one of the key systems, independent of Earth infrastructure, for ensuring safe and resilient mission architectures. Additionally, detecting, localizing and determining the orbits of other space objects around is an important capability for space situational awareness (SSA). This work proposes an integrated methodology that combines optical navigation with anomaly detection to enable resilient spacecraft operations and space situational awareness. Spacecraft position is first determined through a region-matching technique, in which detected lunar surface features are cross-referenced with a catalog generated. Annotated coordinates provide the basis for image-based navigation. Once the spacecraft state is determined, imagery is further analyzed for anomalies indicative of other space objects (natural or artificial). A dual-camera system supplies depth information, and the Herrick–Gibbs method is proposed to reconstruct trajectories from projected surface points observed at multiple epochs. Machine learning methods enhance both navigation and anomaly detection. Convolutional Neural Networks (CNNs) are employed to analyze imagery of lunar regions that are heavily cratered or obscured by shadows, as well as to identify anomalies. A key difficulty in this task is the variability of illumination and the presence of dynamic shadows across the surface. To address this, the CNNs are trained on a region catalog derived from digital elevation models generated in Blender. The training process uses a triplet-based approach with triplet loss minimization to optimize region detection. For accurate matching, a sliding window is applied to align raw images with their corresponding catalog regions. For improved accuracy and contextual awareness, a Shifted Window Transformer (SWIN Transformer) is proposed as a complementary transformer-based approach. For anomaly detection, the Mask Region-based CNN (Mask R-CNN) is employed to perform instance segmentation and classification. A classical instance segmentation approach is applied to identify artificial satellites. Additional disruptive factors, such as natural objects like meteoroids, are detected by analyzing pattern, texture, and color distortions on the lunar surface imagery.

Keywords: Deep Metric Learning, Triplet Loss, Convolutional Neural Networks, Object Detection, Terrain Relative Navigation, Selenodesy

1. Introduction

Terrain-relative navigation (TRN) remains a major challenge in planetary exploration, with numerous techniques proposed for lunar crater recognition. Classical geometric methods, such as [1], [2] and [3] typically detect craters, construct triangular configurations, and then match their descriptors against a reference catalog. While conceptually simple, these methods face significant drawbacks, such as an increase in the number of possible triplets, getting nearly identical geometric patterns in different regions, and compromised reliability of angle- or invariant-based descriptors.

Embedding-based Deep Metric Learning methods have recently emerged as an alternative. Inspired by the FaceNet project [4], where faces captured under varying illumination and viewpoint conditions are embedded into a discriminative space through triplet loss, embeddings can also be applied to lunar surface regions. This enables the model to learn features that remain robust under changes in illumination, shadowing, or rotation. In this study, the FaceNet-inspired triplet-based Deep Metric Learning approach is adapted to region-based navigation and combined with a sliding-window strategy, allowing cataloged regions to be recognized and localized directly in raw imagery.

Comparable deep metric-learning techniques have been applied beyond face recognition. For instance, [5] employs a triplet-loss embedding on spatial map data to distinguish highly similar neighboring regions, enabling tasks such as location prediction and movement-pattern analysis.

In contrast to traditional geometric methods [1, 2, 3], the proposed approach employs a triplet loss function to train a CNN specifically for lunar region matching. This approach bridges the gap between conventional geometric TRN methods and embedding-based Deep Metric Learning. The integration of these techniques reduces ambiguity, addresses scale and perspective challenges, and establishes a robust framework for accurate region identification under demanding imaging conditions.

This study focuses on the determination of the position and attitude states of a satellite orbiting the Moon using only a camera and a gravity sensor. In typical spacecraft operations, several sensor components are used concurrently, with their data fused through estimation filters and related extensions. However, the integration of additional sensors and sensor fusion techniques lies beyond the scope of this work. The gravity sensor is used to measure the roll and pitch angles. Once these angles are obtained, the mechanism on which the camera is mounted stabilizes the satellite, ensuring that the camera always faces the center of the Moon. This action stabilizes the satellite in orbit to perform the presented navigation approach using camera data with minimal deviation. Continuously facing the center of the Moon allows an accurate calculation of the satellite's latitude and longitude. The camera uses an artificial intelligence (AI)-based process to detect special regions in the viewed surface and match these regions with catalog regions. Once the catalog matches are made, the latitude and longitude information of these regions captured in raw photos becomes accessible.

Using the locations of at least three regions, latitude and longitude lines are first overlaid on the entire surface within the camera frame. This allows for the determination of the satellite's yaw angle. Subsequently, once the latitude and longitude information of the point at the center of the camera frame is found, the precise location of the satellite projected on the Moon can be determined.

After the satellite's position and attitude are fully determined, the detection and localization of objects passing in front of the camera system are examined. These objects are defined as "anomalies" because they appear in between the camera and the Moon's surface, which causes color/texture/pattern distortions in the image. For space object detection in space observation, [6] employs instance segmentation with fine-tuned CNN backbones, showing how deep models can effectively handle partially labeled space object data. In this study, a similar process is applied for the detection of artificial satellites.

At least two cameras are used for the object localization process to obtain the depth perception for determining the positions of these anomalies. An AI-assisted process may also be required for anomaly classification, if necessary. Once the positions of anomaly objects are determined over a certain time interval, general or local orbit analyses can be performed for anomalies based on this information.

2. Convolutional Neural Network-Based Region Matching

In this approach, camera images are analyzed to identify distinctive lunar surface regions that have been predefined in a catalog annotated with latitude and longitude information. When such regions are detected, they are matched with their catalog counterparts, and the associated geolocation data is assigned point-wise to the corresponding areas in the image. The catalog is constructed from visually distinguishable regions, each manually selected and cropped into equally sized square patches. Each patch is labeled with its latitude and longitude, corresponding to the center of the cropped area. Ensuring accurate detection, cropping, and matching of these regions is critical to achieving high-precision navigation.

The lunar surface mostly consists of areas that are inefficient for optical navigation. This is because most regions have a uniform color palette. Additionally, the surface features of the Moon often exhibit high similarity, particularly in neighboring regions, which makes distinguishing between them challenging. Neighboring regions generally look like each other. Thus, it is mostly difficult to visually distinguish them in map projects [5]. The densely cratered areas on the far side of the Moon are greatly affected by the position of the Sun and, consequently, the angle of incoming light, causing their appearance to change significantly throughout the day. Particularly, images taken during sunrise and sunset show crater shadows that differ marginally from those taken at noon. This situation led us to use AI support to recognize regions independently of the time of the image taken. Traditional image matching tools (feature descriptors) such as SIFT, SURF, and ORB are vulnerable to extreme points of view (POV) changes and extreme illumination conditions [7]. Besides, their performance can decrease under different rotations. Satellite camera images can be taken from any angle and under any lighting conditions and are also prone to various optical distortions in real missions. As a result, there are two critical challenges to achieving highly accurate optical navigation. The first is that the regions on the surface appear at different rotations depending on the camera's yaw angle. The second is that the regions can be exposed to all kinds of extreme light intensity and direction. As a solution, a CNN-based triplet analysis similar to that used in Google's FaceNet project is proposed.

In the FaceNet project [4], many facial images of different individuals were collected, with each individual's face photographed under various conditions. Each person's face was photographed under different conditions such as full light, half shadow, and from different POVs. Despite being taken from different angles and subjected to various shading, the faces in these images can be matched with their owners through a trained CNN model. The faces are first grouped into triplets. A triplet consists of one anchor sample, one positive sample, and one negative sample. Two different images of the same individual (e.g., one half-shadowed and one fully lit) are assigned as anchor and positive samples, respectively. A face belonging to another person is designated as the negative sample in this triplet. The goal is to learn the common features between the anchor and positive samples through the neural networks, bringing these two faces closer together and distancing the negative sample from them in embedding space. Thus, images of the same person taken under different conditions can be matched with each other, even if they contain significant differences. When encountering a face belonging to someone else, a distinction can be made even if it is photographed under similar conditions.

2.1. Augmentation of Dataset

In this study, the algorithm used in the FaceNet project is combined with the sliding window method to analyze and match the regions. First, the images obtained from the simulated satellite in Blender are subjected to an augmentation process to increase the rotational variation in the dataset. The rotation process is performed in fixed 30-degree intervals for each region. This means that each region has 11 different versions with different rotations. The reason for using a 30-degree interval in augmentation is to prevent the model from "memorizing" due to the high similarity of images at very small angle differences. This issue is discussed in more detail in later sections. Augmentation is only applied for rotation variations, while different angles of light and shadings are included in the different images taken from the Blender environment. This is because accurately reproducing the shadows cast on craters by varying solar illumination angles within the Blender environment is essential for realism, as opposed to applying shading based on a fixed augmentation rule. In total, the augmentations generated through rotations and variations in light direction are combined and organized into triplets for training. See Figures 1 and 2.

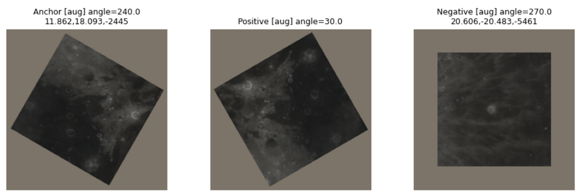

Figure 1: From left to right, an example of a triplet with Anchor-Positive-Negative (APN) samples. The only difference between the anchor and positive is rotation.

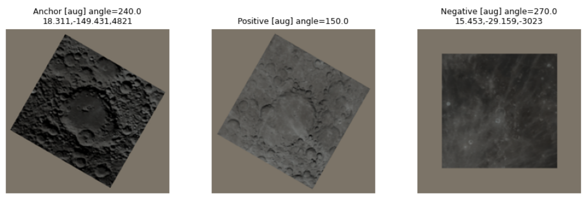

Figure 2: A triplet example with anchor, positive, and negative samples, respectively, from left to right. There is difference in both rotation and the angle of incidence between the anchor and positive samples.

As seen in Figures 1 and 2, there is a brown-gray fill around the rotational augmentations. This fill is added to ensure that the rotated regions do not have their corners cropped. All images are reduced in size and rotated at the center to ensure that the regions are fully within the frame from every angle. The remaining empty parts are filled with this gray-brown color, which is the average color of all images. The reasons for this choice are listed below.

1) When approximating a region with a single fixed value, the value that minimizes the error under the L2 criterion is the mean. This reduces the square error and thus the artificial contrast/edge energy at the fill-content boundary, limiting the unnecessary triggering of early convolution filters in the CNN. Let the pixel values in a crop be $x_1, \ldots, x_N$ and the constant color/value to be filled be c. The constant c that minimizes the L2 (least squares) error solves the following problem:

$$\min_{c\in\mathbb{R}} \sum_{i=1}^{N} \big(x_i - c\big)^2. \tag{1}$$

By setting the derivative to zero, the solution becomes,

$$c^* \;=\; \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{2}$$

x_i: Value of the i-th pixel in the crop. For grayscale, $x_i \in \mathbb{R}$; for color images, it is applied per channel $(x_{i,r}, x_{i,g}, x_{i,b})$.
N: Number of pixels in the crop (per channel).
c: Constant value (color) chosen for filling. For color images, c_r, c_g, c_b per channel.
min: Operation to minimize the error according to the L2/MSE criterion.
c^*: Optimal constant fill; arithmetic mean.

For RGB,

$$\begin{aligned} &\min_{c_r,c_g,c_b} \sum_{i=1}^{N} \sum_{k\in\{r,g,b\}} (x_{i,k} - c_k)^2 \\ &\quad\Rightarrow\; c_k^* = \frac{1}{N}\sum_{i=1}^{N} x_{i,k}, \qquad k \in \{r,g,b\}. \end{aligned} \tag{3}$$

2) Neutral effect with input normalization: If the inputs are normalized per channel as $\frac{x-\mu}{\sigma}$, and the fill color is chosen as the dataset mean μ, the filled pixels will be close to zero after normalization. Thus, these areas provide a "neutral" signal to the model and reduce the risk of interfering with the content.

For an image $I : \Omega \to \mathbb{R}^C$ (pixel position $p \in \Omega$, number of channels C), given the dataset mean $\mu \in \mathbb{R}^C$ and standard deviation $\sigma \in \mathbb{R}^C$ (component-wise $\sigma_k > 0$), the component-wise normalization is:

$$\mathcal{N}(I)(p)_c \;:=\; \frac{I(p)_c - \mu_c}{\sigma_c}, \qquad c = 1, \ldots, C. \tag{4}$$

Fill selection rule. After rotation/transformation, the fill region $\Omega_{\text{pad}} \subseteq \Omega$ is filled with a constant color $c \in \mathbb{R}^C$:

$$I(p) \;=\; c, \qquad \forall\, p \in \Omega_{\text{pad}}. \tag{5}$$

"Neutral" fill condition. If the fill color is chosen as the dataset mean ($c = \mu$),

$$\forall\, p \in \Omega_{\text{pad}} : \quad \mathcal{N}(I)(p) \;=\; \frac{\mu-\mu}{\sigma} \;=\; \mathbf{0} \in \mathbb{R}^C. \tag{6}$$

Thus, after normalization, the fill pixels are the zero vector. The result (effect on the first convolution layer) is below. Let the first convolution weights be W and bias be b. In the fill neighborhood,

$$\psi(p) \;=\; \big(W * \mathcal{N}(I)\big)(p) + b \;=\; \big(W * \mathbf{0}\big)(p) + b \;=\; b, \tag{7}$$

i.e., only the bias contribution remains; the excitation derived from the input is zero.

2.2. CNN Structure

The created triplets are fed into a deep CNN structure in batches. Each batch can contain dozens or, if the processor power is high, hundreds of triplets. CNN, especially for grid-structured data such as images, capture local patterns with convolution filters and hierarchically combine these features across layers to learn tasks such as classification and detection. In this study, the ResNet50 is used as the CNN architecture. This architecture consists of five stages, each containing a different number of consecutive convolutions. In total, it has 49 convolution layers and one fully connected layer, hence the number 50 in its name. As with any CNN model, operations such as pooling layers, activation functions, and flattening are performed in a certain order [8].

The term "deep" in deep learning comes from the multiplicity of these layers. The successive convolutions applied to the visual input delve into the deepest details of the image, helping to extract features. Early layers learn low-level features such as edges, textures, and simple shapes, while deeper layers produce high-level representations such as object parts and semantic components.

2.3. Embedding Production

The output from the deep layers of ResNet50 is passed through L2 normalization. This normalizes the vectors output by the CNN to unit length by dividing them by their own L2 norm.

$$\tilde{z} = \frac{z}{\|z\|_2}. \tag{8}$$

$\tilde{z}$: Normalized version of vector z; for $z \neq 0$, $\|\tilde{z}\|_2 = 1$.
$\|z\|_2$: Euclidean norm, defined as $\|z\|_2 = \sqrt{\sum_{i=1}^{n} z_i^2}$.

This process removes the magnitude information of each vector, leaving only the directional information. This makes the vectors more compact and lower-dimensional, effectively turning the image into a "fingerprint". These compact structures, called embeddings, are feature vectors that preserve the important characteristics and semantic meanings of high-dimensional visual data (pixels) while reducing their dimensionality [9]. In this study, thousands of pixel values are reduced to a 128-dimensional vector. When critical information is stored in these low-dimensional structures, comparing and matching regions on the lunar surface becomes much more computationally efficient and faster.

2.4. Model Training with Triplet Loss Minimization

Triplet loss minimization is one of the most critical elements in model training. It ensures that the distance between the positive sample and the anchor sample in the created triplets is greater than the distance between the negative sample and the anchor sample by a certain margin in embedding space [10]. This process can be called training an embedding-based CNN model by minimizing triplet loss.

An embedding function $f(x) \in \mathbb{R}^d$ with $d = 128$ is produced. After L2 normalization, the embedding has unit Euclidean norm for all inputs:

$$\|f(x)\|_2 = 1, \qquad f(x) \in \mathbb{R}^{128}. \tag{9}$$

Triplet margin inequality can be shown as below:

$$\|x_i^a - x_i^p\|_2^2 + \alpha \ < \ \|x_i^a - x_i^n\|_2^2, \forall\, (x_i^a, x_i^p, x_i^n) \in \mathcal{T}.[4] \tag{10}$$

$f(x)$: A function that maps an image to a d dimensional Euclidean space, $f : \mathcal{X} \to \mathbb{R}^d$.
x_i^a, x_i^p, x_i^n: Anchor (a) and positive (p) samples belonging to the same region; negative (n) sample belonging to a different region. The subscript i represents the identity or triplet index.
α: Margin hyperparameter; forces the anchor-positive distance to be at least α smaller than the anchor-negative distance.
$\mathcal{T}$: The set of all triplets used in training.
$\|\cdot\|_2$: Euclidean norm; $\|z\|_2 = \sqrt{\sum_k z_k^2}$.
$\|\cdot\|_2^2$ denotes its square.

The method employs an embedding function $f_\theta(x)$. The output is L2-normalized, so that all embeddings lie on the unit hypersphere. At the beginning of training, the network is not yet specialized to the target regions and may fail to reliably place positives closer than negatives. During training, for each mini-batch of N triplets (x_i^a, x_i^p, x_i^n), the triplet loss is computed and θ is updated accordingly.

Let the L2-normalized embedding be $\hat{z}(x) = f_\theta(x)/\|f_\theta(x)\|_2$. For each triplet (x_i^a, x_i^p, x_i^n), the loss is

$$\ell_i = \Big[\, \|\hat{z}(x_i^a) - \hat{z}(x_i^p)\|_2^2 \ - \ \|\hat{z}(x_i^a) - \hat{z}(x_i^n)\|_2^2 \ + \ \alpha \Big]_+, \qquad [t]_+ = \max(0, t). \tag{11}$$

See [4].
The mini-batch loss is

$$L \ = \ \frac{1}{N} \sum_{i=1}^{N} \ell_i.[4] \tag{12}$$

The model learns to produce outputs that minimize this loss function. The common features of images of the same region taken under different conditions are identified and learned, and matching operations are performed based on these common features. In other words, with each epoch, the model learns to identify and analyze the critical and distinctive features of the regions. After several epochs, the model produces embeddings that highlight the critical features of the regions, yielding improved matching accuracy.

2.5. Taken Precautions Against Underfitting and Overfitting

One of the most important considerations in model training is ensuring that the model learns sufficiently to generalize, rather than merely memorizing the data. If the anchor and positive samples in the triplets are chosen to be very similar, and the negative is obviously distant, the model may underfit, meaning it does not learn sufficiently. To prevent this, a smart method is used to select triplets, similar to the FaceNet project. CNNs that trained randomly, achieved a correct matching rate of between 60% and 80%. Especially in images taken under challenging conditions (sunrise images, images taken from different points of view rather than directly facing the center of the Moon), the accuracy rate dropped to around 60%, which is insufficient for a critical task such as satellite navigation. The similarity between regions on the lunar surface is much higher than the similarity between faces in the FaceNet project, increasing the risk of underfitting in this study. Furthermore, as mentioned in a similar study using embeddings on maps, the most similar regions are located close to each other in certain areas [5]. The fact that the most similar regions are neighboring regions is one of the biggest challenges in this project. Indeed, considering that matching will be done using the sliding window method, the importance of this problem becomes clear. Therefore, it is necessary to apply the hard positive mining and semi-hard negative mining methods used in the FaceNet project effectively in this study.

In this method, the model identifies regions with embeddings close to each other from each batch. The embeddings of different but similar regions being close to each other negatively affects the accuracy of the matches. It should be remembered that for maximum accuracy in matching, the embeddings of all images of a region must be closer to each other than to the embeddings of all other images of different regions. Regions with high similarity are selected in the same triplet in the next epochs' triplet selections. This way, the model learns the small distinguishing details of similar regions better in the next step, improving performance.

The concept pairs of hard positive and hard negative are discussed in more depth. If the distance between the anchor and the farthest positive is greater than the distance between the anchor and the closest negative, this positive is called a hard positive, and this negative is called a hard negative. Another concept is "semi-hard negative" notion. Here, the distance between the hard positive and the anchor is less than the distance between the negative and the anchor, but by an amount less than a certain margin value. Note that in this case, the positive sample is closer to the anchor as it should be. However, the difference between the anchor's distance to the positive and its distance to the negative being less than the determined margin poses a risky situation. The negative is called as semi-hard negative in this case.

Distances between anchor-positive d_{ap} and anchor-negative d_{an} are,

$$d_{ap} = \left\|\hat{z}(a) - \hat{z}(p)\right\|_2, \quad d_{an} = \left\|\hat{z}(a) - \hat{z}(n)\right\|_2. \tag{13}$$

Mathematical summary of the concept can be shown as follows, also see Figure 3 for visualization.

$$\begin{aligned} \text{Easy negative:} \quad & d_{an}^2 \geq d_{ap}^2 + \alpha, & (14)\\ \text{Semi-hard negative:} \quad & d_{ap}^2 < d_{an}^2 < d_{ap}^2 + \alpha, & (15)\\ \text{Hard negative:} \quad & d_{an}^2 \leq d_{ap}^2. & (16) \end{aligned}$$

However, directly creating a triplet structure with the hard negative and hard positive can cause the model to collapse, especially in the early stages of training. In this scenario, there is a high probability of encountering completely incorrect matches. Especially in monotonous regions like the lunar surface, hard positives and hard negatives can lead to very poor results. Making triplet selections in the later epochs with hard positives and semi-hard negatives yields much healthier results. These hard and semi-hard pairs prevent the model from underfitting by being constantly exposed to easy triplets and from collapsing by being constantly exposed to very difficult triplets. Maintaining this balance is of critical importance [4]. Using semi-hard negative mining and training with hard positives, matching accuracy improved to 92% on noon images and 83% on sunset images. The model remains error-prone under long-shadow conditions.

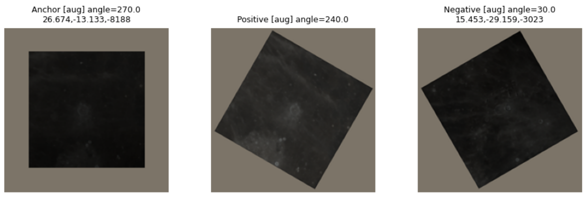

Figure 3: An example of a triplet consisting of anchor, hard positive, and semi-hard negative.

If the distance between the positive and the anchor samples is too small, the model may overfit by memorizing local image characteristics rather than learning discriminative features. This reduces its ability to generalize across different conditions and leads to degraded performance under distributional shifts. Therefore, the dissimilarity between the images of a given region should exceed a minimum threshold to ensure that the learned representations remain robust and transferable. If a large number of very similar images are used (e.g., images taken at 5-degree rotation intervals or images taken under similar lighting conditions one hour apart at noon), the model may also face overfitting. That is, instead of learning the critical features of the regions, it may directly memorize the region. In this case, a memorizing model cannot correctly match an image taken at sunset with heavy shadows or at a very different rotation. Therefore, the dataset was fed with images taken at sunrise, noon, and augmented with 30-degree rotations, all mixed in a random manner.

2.6. Region Matching with Sliding Window Method

When raw images are received from the onboard camera, the satellite computer processes them in a systematic manner. Since all regions in the catalog are pre-labeled as equally sized square patches, a sliding-window approach is applied: a fixed-size square window is moved across the image at small intervals. At each step, an embedding vector is extracted from the windowed image patch using the trained CNN model. The output from the CNN is passed through L2 normalization, resulting in a 128-dimensional vector in the embedding space. The cosine similarity is calculated and compared between this vector and the vectors of all regions in the catalog. The highest similarity is accepted as the match. See Figure 4. A region in the catalog is not assigned to more than one match location; it is only assigned to the highest one. There are two primary challenges associated with this approach.

1) Window size dependency on altitude: Because the camera's intrinsic parameters are fixed, the projected size of a cataloged region on the raw image varies with the satellite's altitude. If the embedding window is too small, it may fail to capture the entire region; if it is too large, it may include irrelevant surroundings. In both cases, the accuracy of region matching degrades, and the window may fail to properly localize the intended region.

2) Centering and alignment: Even with an appropriately sized window, the region of interest may not align with the center of the window. See Figure 5. Since the catalog's latitude and longitude labels are defined with respect to the region's geometric center, any misalignment between the window center and the actual region center in the image leads to positional errors. Such offsets reduce navigation accuracy. The wrong latitude and longitude values may be mapped to the observed pixels, resulting in deviations in the estimated satellite position, as elaborated in the following section.

To solve the window size problem mentioned in the first point, it is proposed to use sliding windows of different sizes one by one over the image and determine the window size based on the best average result. Initially, a relatively large window is slid over the image, and the average value of the similarity rates of the best 20 catalog matches of all embeddings is taken. Then, this process is repeated for smaller window sizes. A matching value average can be obtained for each window size. As a result of this process, the size with the best average is found to be the best size for that orbit. This process took 5-6 minutes in the tests conducted. If the satellite in question is not a satellite whose altitude changes very rapidly, waiting 5-6 minutes to determine the window size makes the method effective because it will use this information for a long time and will not need to repeat it frequently. This process may need to be repeated every few hours or a few times a day, depending on the rate of altitude change.

The centering problem mentioned in the second point arises from several reasons. The first is that the window's sliding interval (stride) is too large, causing the window and the region not to align perfectly. Reducing the window's sliding step can solve this, but it significantly increases the number of embeddings to be extracted, making it an ineffective method. Instead, a denser scan can be performed with a much smaller sliding interval in four directions around the area where the match occurs. This solution focuses the processor's performance on the right place and provides an effective solution. Another reason for this problem, as mentioned earlier, is the high similarity of the regions in an area. This can cause the embeddings of the sliding windows to be matched incorrectly. To prevent this, a transformer-based solution such as the SWIN (Shifted WINdows) transformer can be proposed. A transformer-based algorithm like SWIN, which operates based on context, can better determine the positions of regions relative to each other. The model, which gains contextual awareness, can be more successful in centering during matching. Studies on this issue are still ongoing.

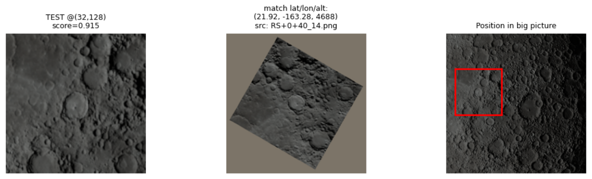

Figure 4: An example of a good match taken at sunset. The left image is a crop from the sliding window. The middle image is the match found in the catalog. The right image is the large image (camera frame) from which the crop was taken.

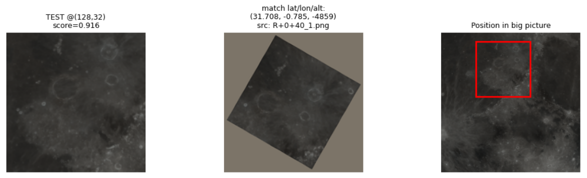

Figure 5: An example of a match with a shifted center. The left image is a crop from the sliding window. The middle image is the match found in the catalog. The right image is the large image (camera frame) from which the crop was taken.

3. Determination of the Full States of the Satellite

In this study, 2-D images acquired by the onboard camera are processed to estimate the satellite's full six-degree-of-freedom (6-DoF) state while orbiting the Moon. This approach falls under TRN, in which navigation is achieved by matching observed terrain features to a reference catalog. Although TRN can be implemented using either cameras or LIDAR sensors, this work employs a camera-only, AI-assisted image processing method to improve the accuracy of position and attitude estimation. While multi-sensor fusion (e.g., camera–LIDAR integration) is often applied in lunar landing missions for higher precision, such extensions are left for future work.

The method is implemented in the Blender simulation environment, focusing on the lunar surface between the latitudes $[-80°, +80°]$. Since the NASA CGI Moon Kit [11] lacks sufficient detail in the polar regions, these areas are excluded from the study. Insufficient data in these regions would not compromise the accuracy of vision-based navigation. If high-resolution maps of the polar regions were available, a geodesic dome structure [12] would be necessary for accurate georeferencing in those areas.

To isolate the performance of the proposed approach, certain conditions are idealized, e.g., the Moon is modeled as a perfect sphere and lens distortions are neglected, as Blender's default camera model does not introduce distortion. These assumptions allow controlled evaluation of the algorithm before introducing additional real-world complexities.

3.1. Assumed Conditions for 6-DoF Determination

The satellite state is defined by six degrees of freedom: three translational components in the Moon-Centered Moon-Fixed (MCMF) frame (latitude, longitude, altitude) and three rotational components (roll, pitch, yaw). In this study, it is assumed that the gimbal mechanism stabilizes the camera in roll and pitch using gravity sensor feedback, keeping the boresight directed toward the lunar center. This nadir-pointing configuration introduces several advantages. First, with roll and pitch set to zero, the image principal point directly corresponds to the sub-satellite latitude and longitude, simplifying position estimation. Second, when the camera views the center of a spherical surface, the projected coordinate grid is highly regular, as latitude lines appear parallel with constant slope, while the central longitude passes through the image center and the remaining longitudes extend symmetrically about it. These geometric properties make the mapping of lunar latitude and longitude from the three-dimensional surface onto the two-dimensional image plane straightforward and accurate. For a satellite in a stable orbit, maintaining nadir pointing via attitude control provides high navigation accuracy with minimal computational effort.

As seen in Figure 6, a frame rendered at (latitude, longitude) = (-47, 90) was generated, and the geographic graticule (latitude/longitude lines) was drawn via Blender scripting to ensure geometric fidelity in that local area. Under the nadir-pointing configuration (camera boresight to the lunar center), all parallels (latitude lines) exhibit identical curvature in the image. The central meridian (the longitude passing through the sub-satellite point) is imaged as a straight line through the principal point, and the remaining meridians appear symmetrically about this line. The image principal point coincides with the sub-satellite point—the projection of the satellite onto the lunar surface—so retrieving the latitude and longitude at this pixel yields the satellite's geodetic latitude and longitude (altitude is estimated separately).

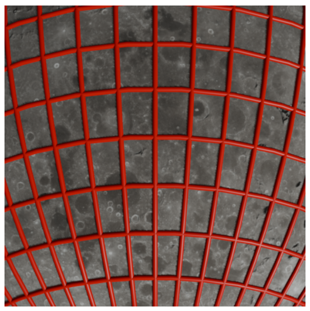
Figure 6: A frame taken from the coordinates (-47, 90) with the corresponding latitude and longitude lines.

We have an image from the camera and the latitude and longitude information of the lunar regions in the image assigned point-wise. In addition, a latitude slope magnitude has been determined for each integer latitude value (-80/+80) within the scope of the project. The latitudes are defined to satisfy the parabolic equation, and the value 'a' in this equation represents the slope magnitude of the parabola. These 'a' values have been tested one by one experimentally, and the ones closest to reality have been selected and manually saved as an array. The values in this array are valid for all spheres and are named as the latitude array.

A similar process has been carried out for the longitude lines. When looking at the latitude values (-80/+80) within the scope of the project from the center of the sphere, the longitude lines are quite close to parabolic lines, as seen in Figure 7. The longitude lines are actually single-armed parabolas with their vertex points on the central longitude line, translated in the direction of the mouths of the latitude parabolas, with a magnitude corresponding to the latitude value at which the image was taken. Here, an array of translation magnitudes has been created experimentally for each integer latitude value (-80/+80) within the scope, by selecting the most suitable ones one by one and arranging them into an array manually. This array is called the longitude array.

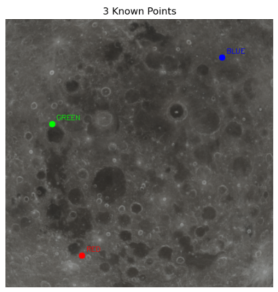

Figure 7: The center points of three regions whose coordinates are determined by AI. The green one is determined as the main point, and the others as assistant points.

3.2. Determination of Latitude, Longitude and Yaw Angle

The satellite's latitude and longitude in the MCMF frame can be determined once the principal point (image center) is geolocated. These coordinates are obtained by referencing the relative positions of at least three cataloged regions detected by the AI model. Each detected region provides a correspondence between its known latitude/longitude and its image-plane location. The assignment of at least three reference points to the image allows for the construction of a local mapping between the image and the lunar surface. While three points are sufficient for a minimal solution, increasing the number of correspondences improves both accuracy and robustness. The method applied in this study is called "rotating central longitude" method. For clarity of exposition, the following description proceeds with the case of three reference points. The steps can be listed as,

1) The point closest to the principal point, i.e., the satellite's projection point on the Moon, is determined as the main point among the three points. The other two points are assigned as assistant points. As seen in Figure 7, the green point is determined as the main point and the others as the assistant points.

2) A line of the form $y = Ax + B$ that will pass through the center point of the frame is defined. The slope A of this line is put into a loop with values between -20 and +20. In each step of the loop, the line takes on a different value of A, and by the end of the loop, it will have scanned the entire image. This line represents the central longitude line on which the satellite is located.

$$y_{\mathrm{cp}} = A_{\mathrm{cl}}\, x_{\mathrm{cp}} + B_{\mathrm{cl}}. \tag{17}$$

A_{cl}: Slope coefficient of the central longitude line.
B_{cl}: Intercept coefficient of the same line (in pixel axes).
x_{cp}, y_{cp}: Pixel coordinates of the frame's center (center point).

3) In each iteration, we construct a parabola whose vertex lies on the central longitude line and whose tangent at the vertex is perpendicular to that line; the parabola passes through the main point, and its slope parameter "a" is taken from the latitude array according to the main point's latitude.

$$y = a\,(x - r)^2 + k. \tag{18}$$

Parabola equation (with the vertex at (r, k)) where the vertex and a point through which it passes are known.

$$k = A_{\mathrm{cl}}\, r + B_{\mathrm{cl}}. \tag{19}$$

The coordinates of the vertex (r, k) lie on the line $y_{\mathrm{cp}} = A_{\mathrm{cl}} x_{\mathrm{cp}} + B_{\mathrm{cl}}$.

$$y_{\mathrm{mp}} = a\,(x_{\mathrm{mp}} - r)^2 + A_{\mathrm{cl}}\, r + B_{\mathrm{cl}}. \tag{20}$$

x_{mp}, y_{mp}: x and y components of the main point.
a: Curvature coefficient of the parabola (taken from the latitude array).

4) Subsequently, longitude lines are determined in each iteration. Their vertices also lie on the central longitude line but are shifted towards the closest polar region by a value taken from the longitude array according to the latitude information of the main point, as previously mentioned. Nearest pole direction can be determined by considering and comparing the determined latitude vertices' positions on the central longitude. Besides, one arm of the longitude parabola must pass through the detected region points. The other arm can be ignored.

5) The same process is repeated for the other two auxiliary points. In each iteration, three latitude parabolas, three longitude parabola arms, and one central longitude line are derived for three distinct points. See Figure 9. The vertex points of the three latitude parabolas on the central longitude line are identified. In each iteration, the pixel distances between the vertex points of the parabolas passing through the auxiliary points and the vertex point of the parabola passing through the main point are measured. The ratios of these distances are then calculated. Additionally, the intersections of the determined longitude arms with the latitude parabola of the main point are identified. The arc distances between these intersections along the main point's latitude parabola are measured, and their ratios are also calculated for further use.

6) After computing the line and arc distances between the latitude and longitude lines of the main point and each auxiliary reference point in the image, their relative latitude and longitude ratios are determined. The same procedure is applied in geographical space, where the true distances between corresponding latitude and longitude values are calculated in kilometers, and their ratios are derived. If the ratios derived from pixel measurements match the ratios derived from real-world distances for both latitude and longitude lines, this confirms that the candidate central longitude line (with parameter A) is correctly aligned with the true central longitude. As shown in Figure 8, the upper panel displays the pixel distances of the latitude parabola vertices of the auxiliary points to the vertex of the main point's latitude parabola, plotted against changes in the 'A' value. The lower panel shows the ratios of these distances, also plotted against the changing 'A' value. The real ratio, indicated by the orange dashed line in the lower panel, must intersect the plot at two distinct 'A' values, resulting in two possible scenarios. With the inclusion of longitude intersections, one of the scenarios give the same angle as latitude intersections and true candidate is determined.

7) Once the real scenario with true "A" value for central longitude line is selected, the slope of the central longitude line in the frame gives the yaw angle.

8) As a result of the first six steps, the latitude and longitude lines will have been drawn to pass through three points. The latitude and longitude values of the center point, i.e., the satellite, can now be determined. The center point is the vertex point of its own latitude parabola and can be determined by taking the ratio of its distances to the vertex points of the other latitude parabolas. Similarly, the

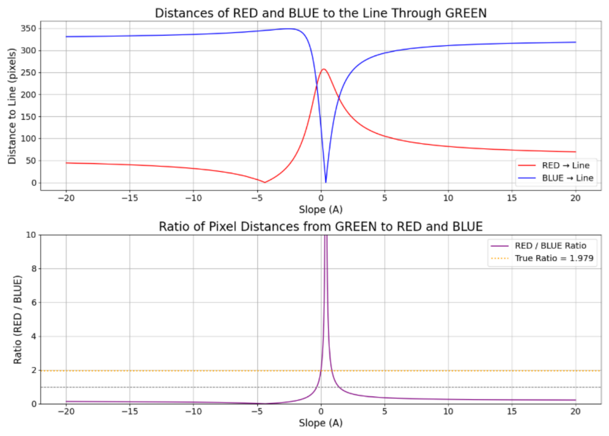

Figure 8: Distance and ratio plots of latitude vertices.

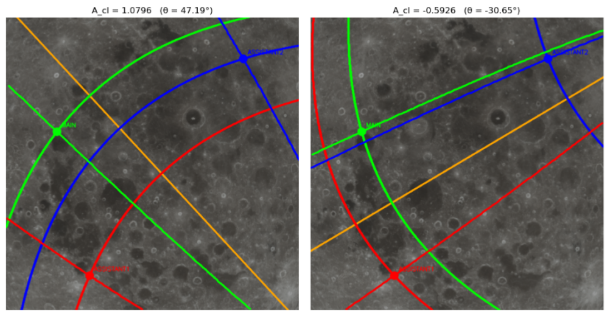

Figure 9: Rotating central longitude method. Central longitude shown as orange line. It is rotating 360 degrees around the center point of the picture.

longitude value of the central longitude line can be calculated based on the longitude values of the other three points. When the arc length distances of the three points to the central longitude line are measured (the distances should be measured along their own latitude parabolas) and their ratios are taken, the longitude value of the central longitude line will also be determined. See Figure 10.

The latitude and longitude lines drawn within the latitude range of -80 to +80 degrees give a maximum deviation of 3-4 degrees. This error arises from the assumption that the latitude of the main reference point closest to the image center is identical to the latitude of the principal point. In practice, the latitude and longitude values at the center are interpolated from predefined arrays using this approximation. Over large intervals, this assumption introduces small discrepancies, particularly when the main reference point lies far from the image center. The deviation can be reduced by applying a numerical refinement procedure, as iteratively updating the

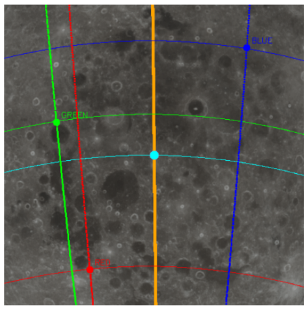

Figure 10: Determined true fit of both latitudes and longitudes. Cyan point at the center represents the satellite's projection point.

estimated center coordinates using the derived values and repeating the mapping steps until convergence.

3.3. Determination of Altitude

At this stage, only the altitude information of the satellite in the MCMF frame remains to be determined. To determine this information, the intrinsic properties of the camera must be known, and the triangulation method can be used.

3.3.1. Camera Structure:

The simplest camera model can be used as the pinhole camera. It can be conceptualized as an empty box with a small aperture on one face. The plane containing the aperture is called the perspective plane, and the aperture itself is the perspective center [13]. The opposite face of the box serves as the image plane, which spans the camera's x– and y–axes, while the z–axis is aligned with the optical axis through the pinhole [13]. Light rays from an external object enter the box through the aperture, and since all rays must pass through this single point, an inverted image of the object is formed on the image plane along both axes.

Real cameras deviate from the ideal pinhole model because they use lenses, which introduce optical distortions (e.g., radial and tangential), most pronounced in fisheye systems. In this study, the Blender camera is used in its default, idealized configuration, which is distortion-free. Although Blender can simulate various distortions to increase realism, we assume an ideal camera and therefore omit distortion calibration.

In image-based terrain relative navigation, the camera's intrinsic parameters are critical. Four intrinsics are required for the ensuing positioning calculations, as the principal point, sensor size, focal length, and image resolution. They will be discussed in detail before introducing the altitude determination.

Principal Point: The principal point is the point where the optical axis intersects the image plane. The optical axis corresponds to the Z-axis of the camera, i.e., the direction in which the pinhole, the perspective center, is looking [13]. The principal point coincides with the exact center of the image, i.e., the center of the image plane, unless there is a shift or distortion caused by the lens. Since the Blender camera is perfect, the principal point will be the center of the image throughout this study. This is equivalent to the (256, 256) pixel position in a 512×512 pixel photo.

Sensor Size: The sensor size is the area of the sensor on the image plane in square millimeters. The sensor of the Blender camera is set to be square, and the side length is defined as 36 millimeters.

Focal Length: The focal length is the distance between the camera's image plane and the pinhole, the perspective center. When the sensor is not square, there are X and Y components [13]. However, since the sensor of the camera used in this study is square, the X and Y components are equal to each other. Operations will be performed based on this equality using a single focal length. In this study, focal length is selected as 50 mm by Blender default.

Resolution: The resolution of the camera is a two-dimensional quantity, similar to the sensor size, and it indicates the number of pixels per unit length along the sensor's edges. The Blender camera has a resolution of 512×512 pixels in both the X- and Y-axes. Given that the sensor size is 36×36 mm, the pixel density is approximately 14.22 pixels per millimeter. This is calculated as follows:

$$\frac{512 \text{ pixels}}{36 \text{ mm}} \approx 14.22 \text{ pixels/mm}.$$

Consequently, each pixel corresponds to approximately 0.07 mm on the sensor, calculated as:

$$\frac{1 \text{ mm}}{14.22} \approx 0.07 \text{ mm/pixel}.$$

This information allows for the conversion of measured pixel distances to real-world distances in millimeters.

3.3.2. Triangulation

Triangulation is the process of finding the position of a third point using the position information of two known points. The three points are triangulated, and the position of the third point is found using triangle similarity/properties. In this study, the altitude information will be found using this triangle and another triangle formed inside the camera, using triangle similarity.

As in other Moon TRN projects, this method is a reliable method frequently used. In a Lunar TRN project [1], navigation is performed by detecting craters and using triangulation based calculations on the positions of the detected craters. In this study, triangulation is instead performed using the positions of specially selected surface regions. Triangulation can generally be categorized into two types: intersection, in which the positions of the camera and one known location are used to determine another location, and resection, in which the positions of two or more known locations are used to determine the position of the camera [1]. The method employed in this work corresponds to resection triangulation.

Steps:

1) As seen in Figure 11, first, the center points of two representative regions whose latitude and longitude information are known are assigned as two points with latitude and longitude information in the MCMF frame. These representative regions' centers are shown as yellow and green spheres. The blue line, which represents the distance between these two points, is mathematically created using the straight-line equation whose two points are known.

$$\mathbf{r}(t) = \mathbf{r}_1 + t\,(\mathbf{r}_2 - \mathbf{r}_1) = (1-t)\,\mathbf{r}_1 + t\,\mathbf{r}_2, \qquad t \in \mathbb{R}. \tag{21}$$

$\mathbf{r}(t)$: position vector on the line at parameter t (e.g., in the MCMF frame).
$\mathbf{r}_1 = [x_1,\, y_1,\, z_1]^\top$, $\mathbf{r}_2 = [x_2,\, y_2,\, z_2]^\top$: position vectors of the two known points.
t: scalar parameter; for the line segment $t \in [0, 1]$, for the infinite line $t \in \mathbb{R}$.

2) Using this blue line in the triangle similarity calculation will cause deviations in many scenarios. Because a point determined on a spherical surface may be much closer to the camera than another point, this line appears on the image plane not in its original size but multiplied by its cosine value. Since the cosine angle cannot be calculated with only the images taken from the camera, another method must be used.

3) The three-dimensional coordinates of the midpoint of the blue line are found from the straight-line's equation, and a faint orange plane is defined on this point, as seen in Figure 11. To define the plane, both a reference point and a direction vector are required. The direction vector is taken as the line from the satellite's projection point on the lunar surface to the Moon's center, referred to here as the camera point-of-view (POV) vector. Thanks to this vector, regardless of how different the distances from the detected points on the Moon's surface to the satellite are, the faint orange plane defined perpendicular to this vector will always be parallel to the image plane of the camera. This construction guarantees the accuracy of the triangle similarity.

$$\mathbf{n} = \mathbf{q}_2 - \mathbf{q}_1. \tag{22}$$

Normal vector obtained from two known points defining the direction perpendicular to the plane. $\mathbf{q}_1$: Satellite's projection point on the Moon (latitude-longitude). $\mathbf{q}_2$: Moon center.

$$\mathbf{n} \cdot (\mathbf{r} - \mathbf{r}_0) = 0. \tag{23}$$

Plane equation; $\mathbf{r}_0 = (x_0, y_0, z_0)$ is a known point on the plane, $\mathbf{r} = (x, y, z)$ is an arbitrary point on the plane.

$$a\,x + b\,y + c\,z + d = 0, \qquad d = -\big(a\,x_0 + b\,y_0 + c\,z_0\big), \tag{24}$$

Coordinate form; $\mathbf{n} = (a, b, c)$ and $\mathbf{r}_0 = (x_0, y_0, z_0)$. $\widehat{\mathbf{n}} = \mathbf{n}/\|\mathbf{n}\|_2$: unit normal

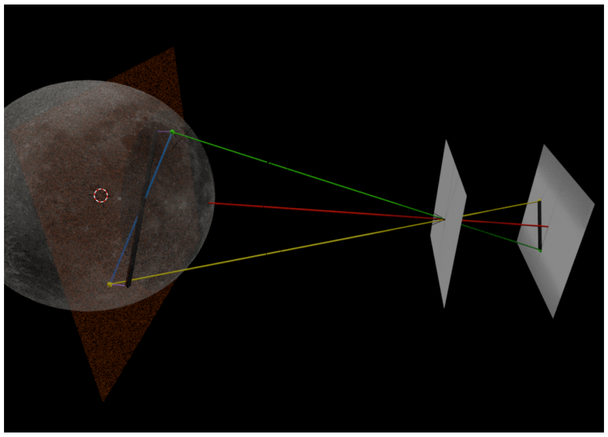

Figure 11: Geometric relationship between two similar triangles: One formed by the camera position and two known surface regions in the MCMF frame, and the other formed within the image plane by the projections of these regions and the principal point.

4) The projection points of the green and yellow spheres representing the two points whose coordinates are known onto the orange plane are determined according to Equation 25. A straight line,

shown in black, is defined between these projection points. Since this line lies on the faint orange plane, which is parallel to the image plane, it will lead to triangulation operations that yield correct results without causing deviations in navigation, even in the most extreme scenarios. That is, no matter how different the distances of the green and yellow points from the camera are, this method will ensure that the pixel distance seen in the camera (the black line on the image plane) and the calculated distance (the black line on the orange plane) are suitable for establishing a triangle similarity relationship.

$$P' = P - \frac{Ax_0 + By_0 + Cz_0 + D}{A^2 + B^2 + C^2} \cdot (A, B, C) \quad (25)$$

$P(x_0, y_0, z_0)$: the coordinates of the point in 3D space,
(A, B, C): the normal vector of the plane,
P': the projection of point P onto the plane.

5) Since the equations of all components mentioned above are established, the distance from the point where the camera POV vector intersects the faint orange plane to the satellite's projection point on the Moon can be measured. Let's call this distance the "orange plane distance". The sum of the altitude and the orange plane distance gives the camera's distance to the orange plane. The ratio of this distance to the focal length is equal to the ratio of the length of the black line on the faint orange plane to the pixel distance represented by the black line on the image plane. As a result of the triangle similarity, the camera's distance to the orange plane can be found. Subtracting the orange plane distance from this distance gives the altitude.

$$\frac{f}{d_p + d_c} = \frac{L_{\text{ibl}}}{L_{\text{obl}}} \quad (26)$$

f: focal length,
d_p: distance between the orange plane and the Moon surface,
d_c: distance between the camera and the Moon surface,
L_{obl}: black line length between two projection points on the orange plane,
L_{ibl}: black line length between two points on the image plane.

4. Anomaly Detection, Localization and Orbit Determination

4.1. Anomaly Detection

Anomalies such as meteoroids or other artificial objects may enter the frame of the satellite's camera while it is in orbit. These objects will appear as anomalies in the satellite camera monitoring the Moon's surface. It is possible to detect these anomalies and determine their locations. To detect and mask these objects in a supervised approach, the Mask R-CNN (ResNet-50-FPN) model on Detectron2 is fine-tuned using an artificial satellite dataset labeled in COCO format. The dataset is generated by rendering artificial objects in cislunar scenes created in Blender with different sizes, colors, positions, lighting and if necessary, motion blur. When all examples are provided with complete masking labels and 'negative' frames, it is possible to reliably extract the location and masks of objects by fine-tuning a COCO pre-trained network. The trained model achieves an 86% accuracy in detecting artificial satellites. Once detected, the satellites are successfully masked at their centers. As a further application, specific artificial satellites orbiting around the Moon can be detected and identified with triplet minimization approach that mentioned in detail.

A similar algorithm exists in a project that distinguishes between natural and artificial objects in space. The study proposes a relabeling architecture consisting of a detector with Faster R-CNN + ResNet-18 backbone on Detectron2, followed by a small CNN classifier that separates point-like, streak, noise, to relabel space objects in partially labeled astronomical images [6].

Anomaly detection can also be performed using generic methods that capture color/texture/pattern inconsistencies to detect undefined objects; the color, texture, pattern disruption-based approach offers both broader coverage and simpler implementation.

As seen in Figure 12, the anomaly is detected based on color/texture/pattern mismatch and its center point is marked. This point indicates where the anomaly falls on the Moon's surface relative to the camera.

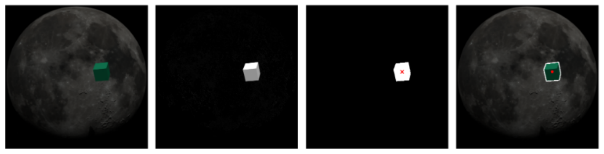

Figure 12: An anomaly detected and masked based on color/texture/pattern disruption, with its center point marked.

4.2. Object Localization

Once the satellite's position is known, a line can be defined between the satellite and the point where the detected anomaly (identified as an object) falls on the Moon's surface according to Equation 21. The location of this point can be determined using the method previously used to determine the satellite's location projected on the Moon's surface. The center point of the red dot in Figure 12 can be determined relative to the positions of at least three other detected regions.

In a multi-camera system (which may consist of multiple cameras on a single satellite or cameras distributed across different satellites), the same surface feature on the Moon can be observed from different viewpoints. Each observation defines a projection ray in three-dimensional space, constructed from the camera position and the corresponding image-plane projection of the feature. The intersection of these rays provides the three-dimensional location of the detected object on the lunar surface.

As seen in Figure 13, the lines formed by the cameras and the projection points intersect at one point, and this point gives the location of the object.

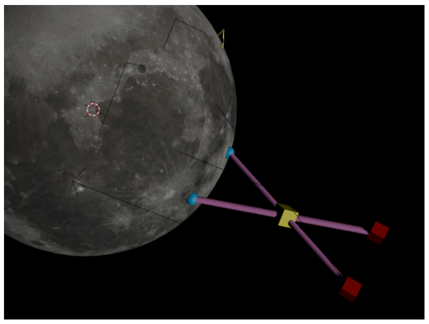

Figure 13: Object localization using two cameras with known positions. Red cubes indicate the cameras, the yellow cube marks the object, and blue spheres show the surface points where each camera observes the object.

If the object's position can be determined with sufficient accuracy at three different observation times, its orbital properties and velocity can be estimated. Prior to applying the Herrick–Gibbs method for velocity estimation, the position vectors must first be transformed from the MCMF frame to the Moon-Centered Inertial (MCI) frame. The Herrick-Gibbs method is based on a short-arc assumption. The method is sensitive to measurement noise; therefore, in practice, it is recommended to refine the results with a least-squares or Extended Kalman Filter (EKF) improvement step using four or more measurements [14]. Additionally, the reliability of the solution can significantly decrease if the geometry of the three points is unfavorable, especially in nearly collinear or highly symmetric configurations.

5. Discussions and Conclusion

In this study, the accuracy of results in three critical applications is of vital importance. First, the accuracy and precision of the results obtained during region matching; second, the accuracy and precision of the coordinates derived from the rotating central longitude method; and third, the precise masking of anomalies and the determination of their center points during anomaly detection. Since all other processes are purely mathematical calculations and sensor deviations are disregarded, their contribution to the error margin is negligible.

In the region detection phase, the accuracy rate varies between 83% and 92%, depending on the intensity of shadows and whether the surface is cratered. This is a highly satisfactory result, showing significant improvement from the previous test results, which ranged between 60% and 80% before applying semi-hard negative mining. However, the model has yet to achieve satisfactory results in precisely cropping regions and accurately distinguishing them from adjacent terrains. Although the matched regions are largely correct, only about 43% of the regions are cropped exactly from their centers. These centering errors are the primary source of deviations in the results. Work on this issue is ongoing, and a transformer-based auxiliary model, such as SWIN, will be studied to enhance contextual awareness. Additionally, incorrect matches can disrupt the calculations of the rotating center longitude method. To mitigate this, a simple filtering mechanism has been developed. The latitude and longitude information of all detected regions on the raw photograph is collected in a pool, and outliers are eliminated. Furthermore, when all detected regions are processed simultaneously, incorrectly matched regions that escape filtering can introduce noise and reduce accuracy. Therefore, instead of processing all regions at once, they are processed in groups of three, and outliers from these results are removed through a second filtering step.

Secondly, in the rotating central longitude method, the deviations in latitude and longitude assignments of detected points are around a maximum of 3-4 degrees. This deviation arises when the main point is far from the center of the frame. Particularly in wide-angle shots of distant terrains, if the main point is selected from the edge, the coefficients derived from the latitude and longitude arrays do not provide an accurate fit for the central point. This issue is more critical for longitude values. Therefore, if the main point is not near the center and the camera frame covers a wide terrain, the results of this method should be approached with an awareness of potential deviations. To minimize these deviations, the method can be repeated with a certain number of iterations under these conditions, and a numerical approximation can be applied. When the main point is close to the center of the frame, the deviation in the detected latitude and longitude angles is around 1 degree. This may be due to the coefficients of the latitude and longitude arrays not being perfectly selected for all conditions.

Finally, in the object detection phase, Detectron2 demonstrated highly satisfactory results by accurately centering and effectively masking the artificial satellites in the images. Similarly, simple tools effectively masked color/texture/pattern disruptions and located their centers. No further improvements were deemed necessary in this area.

In conclusion, this study demonstrates a successful application of AI-enabled optical navigation and anomaly detection for spacecraft in lunar orbit. By integrating traditional geometric TRN methods with deep metric learning techniques, a robust framework is established even under challenging imaging conditions.

Acknowledgments

The first author was supported by the TUBITAK 2224-A International Scientific Meetings Fellowship Programme, and would like to express their gratitude to TUBITAK for the financial support. Additionally, the first author would like to extend special thanks to Burak Ay for his moral support and inspiration throughout this study.

References

[1] Ava C Thrasher, John A Christian, Giovanni Molina, Michael Hansen, John Y Pelgrift, and Derek S Nelson. Lunar crater identification using triangle reprojection. *Proceedings of the AAS/AIAA*, 2023.

[2] John A Christian, Harm Derksen, and Ryan Watkins. Lunar crater identification in digital images. *The Journal of the Astronautical Sciences*, 68(4):1056–1144, 2021.

[3] Roberto Del Prete and Alfredo Renga. A novel visual-based terrain relative navigation system for planetary applications based on mask r-cnn and projective invariants. *Aerotecnica Missili & Spazio*, 101(4):335–349, 2022.

[4] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[5] Vincent Spruyt. Loc2vec: Learning location embeddings with triplet-loss networks.

[6] Florin Dumitrescu, Bogdan Ceachi, Ciprian-Octavian Truică, Mihai Trăscău, and Adina Magda Florea. A novel deep learning-based relabeling architecture for space objects detection from partially annotated astronomical images. *Aerospace*, 9(9):520, 2022.

[7] ISIK Murat. Comprehensive empirical evaluation of feature extractors in computer vision. *PeerJ Computer Science*, 10:e2415, 2024.

[8] Norah A. Al-Humaidan And and Master Prince. A classification of arab ethnicity based on face image using deep learning approach. *IEEE Access*, 9:50755–50766, 2021.

[9] Raia Hadsell, Sumit Chopra, and Yann Lecun. Dimensionality reduction by learning an invariant mapping. pages 1735 – 1742, 02 2006.

[10] Byungsoo Ko and Geonmo Gu. Embedding expansion: Augmentation in embedding space for deep metric learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7255–7264, 2020.

[11] NASA Scientific Visualization Studio (SVS). Cgi moon kit (id 4720), 2019. Color map adapted from LROC WAC Hapke-normalized mosaic; displacement map from LRO LOLA gridded products. Last updated 2025-01-06.

[12] Kin Lei, Dongxu Qi, and Xiaolin Tian. A new coordinate system for constructing spherical grid systems. *Applied Sciences*, 10(2):655, 2020.

[13] Kuan-Ying Lin, Yi-Hsing Tseng, and Kai-Wei Chiang. Interpretation and transformation of intrinsic camera parameters used in photogrammetry and computer vision. *Sensors*, 22(24):9602, 2022.

[14] Howard D Curtis. *Orbital mechanics for engineering students.* Butterworth-Heinemann, 2019.