Acta Astronautica 245 (2026) 430–460

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Research paper

Development and validation of a physically based rendering methodology for celestial bodies

Andrea Pizzetti [a,*,1], Paolo Panicucci [a,2], Francesco Capolupo [b,3], Francesco Topputo [a,4]

[a] Politecnico di Milano, Via La Masa 34, Milan, 20156, Italy
[b] European Space Agency (ESA), Keplerlaan 1, Noordwijk, 2201 AZ, Netherlands

ARTICLE INFO

Keywords:
Physically-based rendering
Celestial bodies
Image formation
Synthetic image
Radiometric validation

ABSTRACT

This article outlines a physically based rendering methodology for generating spectral images of planets, moons, and other quasi-spherical bodies. The methodology links physical radiometric quantities such as the collected power or the electron flux to each pixel of the synthetic image. Conversely to standard ray-tracing methods, the image is formed by discretizing the spherical body into longitude–latitude quadrangles, which are projected into the camera and directly gridded to the detector pixels. Despite the high number of sectors undergoing this procedure, smart bounding, efficient sampling, and pruning of the inactive sectors enable fast and efficient computations. Local radiometric and geometric parameters are interpolated from texture maps to enhance the fidelity of lighting and shadows. Different reflection models can be used and mixed together. The output of the renderer is a radiometrically-consistent image reproducing the exact interaction between the light, the object in the scene, and the camera. The methodology is validated against analytical laws and real Moon images acquired in orbit and from the ground. Results reveal very good consistency in luminance, contrast, and structure. The frame rate for a Moon flyby scenario spans over 10 Hz at far range to about 1 Hz at a closer range, consistent with current limits in CPU-based renderers. Thanks to its radiometric consistency, the methodology proves effective for tuning the camera exposure time brackets and is essential for the operation of hardware-in-the-loop optical stimulators. A tool implementing the methodology is released to the public as an open-access rendering application.

1. Introduction

The physical consistency of the data used for algorithm design, model training, and performance verification is of utmost importance in most of the engineering sectors. *Domain gap* refers to the differences that can be found between the design and the deployment environments, which lead to degraded performance in real-world applications. Within the space sector, this issue is even more impactful given the inherent high risks and costs involved in the missions.

Unfortunately, space is characterized by a huge domain gap, mainly stemming from the inherent scarcity, low quality, and small variability of ground truth data. At the same time, there is a strong push nowadays towards the use of data-driven algorithms, which rely on vast amounts of diverse and high-quality data to grant generalization and performance. Notably, these two aspects conflict with each other.

When speaking of optical sensors and images, the consequences of the domain gap may affect the performance of vision-based algorithms or even impair the scientific results of the mission. For example, pose estimation algorithms trained on synthetic images and tested against real images showed drops in performance due to differences in visual features and illumination conditions [1,2]. The long-range autonomous rendezvous experiment of the PRISMA mission suffered from non-linear unmodeled optical effects (e.g. *blooming*) impacting the inertial and relative navigation solution [3]. This was also observed in the context of the STAR Nav study, which assessed navigation solutions at mid-range distances from the Moon using a high-Technology Readiness Level (TRL) star tracker [4]. During the outbound powered flyby of Artemis I, the method used to determine camera exposure settings, purely based on phase angle variations, resulted in heavily overexposed images due to the non-linearity of the brightness behavior at close

* Correspondence to: Department of Aerospace Science and Technology, Italy.
E-mail address: andrea.pizzetti@polimi.it (A. Pizzetti).
[1] Ph.D. Student.
[2] Assistant Professor, Department of Aerospace Science and Technology.
[3] GNC Systems Engineer, Guidance, Navigation and Control Section.
[4] Full Professor, Department of Aerospace Science and Technology.

https://doi.org/10.1016/j.actaastro.2026.03.003
Received 4 July 2025; Received in revised form 25 February 2026; Accepted 4 March 2026
Available online 6 March 2026

Nomenclature

P	Power W
M	Exitance $\mathrm{W\,m^{-2}}$
I	Intensity $\mathrm{W\,sr^{-1}}$
F	Flux Density $\mathrm{W\,m^{-2}}$
E	Irradiance $\mathrm{W\,m^{-2}}$
L	Radiance $\mathrm{W\,sr^{-1}\,m^{-2}}$
Ω	Projected solid angle sr
ω	Solid Angle sr
p_G	Geometric Albedo –
p_N	Normal Albedo –
α	Phase Angle rad
λ	Wavelength m
(P/L)	Power-to-Radiance $\mathrm{sr\,m^2}$
S	Signal $\mathrm{e^-\,s^{-1}}$
F_{eff}	Effective Flux Density $\mathrm{W\,m^{-2}}$
f	Focal Length m
μ	Pixel Size m
c	Optical Center px
s	Pixel Skew px
Δt	Exposure Time s
G_{AD}	Analog-to-Digital Gain $\mathrm{DN\,e^{-1}}$
QE	Quantum Efficiency –
T	Transmittance –
$\mathrm{DN}^{(0)}$	Offset DN
A	Area $\mathrm{m^2}$
R	Radius m
D	Displacement m
d	Distance m
ϵ	Offpointing Angle rad
t	Epoch s
T	Temperature K

Functions

f_r	BRDF $\mathrm{sr^{-1}}$
F_r	BRDF Integral –
Ψ	Phase Law –
$\mathcal{F}$	Gridded Interpolant

Frames

O	Object-Fixed Frame
B	Body-Fixed Frame
C	Camera-Fixed Frame
CSF	Camera-Sun Frame
$A_{x\to y}$	Direction Cosine Matrix

Vectors

$\boldsymbol{r}$	Position m
$\hat{\boldsymbol{n}}$	Normal Direction –
$\hat{\boldsymbol{s}}$	Body to Sun Direction –
$\hat{\boldsymbol{c}}$	Body to Camera Direction –
$\hat{\boldsymbol{b}}$	Camera Boresight –

Coordinates

ϕ	Azimuth rad
θ	Coelevation rad
Φ	Longitude (CSF) rad
Θ	Colatitude (CSF) rad
Λ	Latitude (CSF) rad
ϕ	Longitude (B) rad
λ	Latitude (B) rad
β	Bearing (C) rad
x	Image Plane Horizontal –
y	Image Plane Vertical –
u	Pixel Array Horizontal px
v	Pixel Array Vertical px

Subscripts

$(\cdot)_e$	Emission
$(\cdot)_i$	Incidence
$(\cdot)_r$	Reflection
$(\cdot)_\lambda$	Spectral
$(\cdot)_j$	Point Index
$(\cdot)_{k,l}$	Sampling Index

Superscripts

$\hat{(\cdot)}$	Direction/Normalized
$\bar{(\cdot)}$	Mean/Midpoint
$\tilde{(\cdot)}$	Processed
$(\cdot)'$	Distorted

distances from the surface [5]. Over-exposure issues also occurred during the Earth certification pass, due to stronger reflections of cloud and ice regions [6]. All these examples show the importance of having a reliable and validated test bed beforehand.

To overcome the domain gap, the space imaging engineering community worked in developing tools that could provide reliable data for mission design and Verification & Validation (V&V) of algorithms. Current state-of-the-art approaches rely on the generation of synthetic data through *render engines* [7–18], *optical stimulators* [19–26], or with cameras attached either to an *air-bearing platform* [27–30] or to *robotic arms* [31–34] imaging a mock-up of the target in a simulated space environment.

This article deals with the first approach, which is also the most common implemented solution. In fact, in the past years, the increasing potential of cameras as low-cost multi-purpose sensors pushed many commercial entities, national agencies, and research institutions towards the financing and development of proprietary render engines for resolved targets, i.e. whose projection on the image spans more than one pixel.

An example of a commercial tool tailored for space applications is SurRender [7,8], developed by Airbus Defence & Space, which uses a proprietary path-tracing implementation to handle multiple diffusions and reflections. It is capable of generating images of celestial objects from far to close range and capturing shadow effects. A fast mode based on rasterization with OpenGL[5] can be used enabling higher frame rates at the cost of physical accuracy. Another example is the Planet and Asteroid Natural Scene Generation Utility (PANGU) [9–11], which has been developed by the University of Dundee under ESA funding as a versatile tool capable of generating optical, thermal, and LiDAR images thanks to OpenGL and programmable Graphical Processing Unit (GPU) shaders. The NAIF/SPICE system is integrated in PANGU, and single-scattering atmospheric models are available.

Looking at the research and academia efforts, the Inter-planetary Rendering for Imaging and Sensors (IRIS) [12] is a render engine

[5] https://www.opengl.org/.

developed by JPL that builds upon the GPU-accelerated OptiX ray-tracing library to grant a very high frame rate, even if the effects of indirect lighting (i.e., soft shadows) cannot be captured. Building upon the physically-based render engine Mitsuba,[6] the Navigation and Rendering Pipeline for Astronautics (NaRPA) [13] from Texas A&M University provides a full pipeline for realistic surface rendering of landing and rendezvous trajectories.

However, these proprietary tools are notably restricted to the general public. This limitation led the community to tailor existing open-source software to their necessities [14,35]. Among these, the Space Imaging Simulator for Proximity Operations (SISPO) is based on the open-source Blender Cycles.[7] The tool includes modeling of comet dust environments [15]. Blender has also been successfully used in many other tools, owing to its simplicity and the accuracy of its rendering core Cycles based on path-tracing. For example, the Celestial Object Rendering TOol (CORTO) [36] offers a powerful solution for generating photorealistic images of celestial bodies, interfacing with the Blender architecture through Python scripts. A Graphical Utility for Icy moon Surface Simulations (GUISS) [18] was also developed at JPL, exploiting a Blender interface for stereo images generation of Europa and Enceladus terrains. Finally, Unreal Engine[8] has been used in NASA's Digital Lunar Exploration Sites Unreal Simulation Tool (DUST) [16] for virtual reality applications targeting surface mobility near the lunar south pole.

These open-source tools may look interesting at first sight, but their validation, efficiency, and radiometric consistency are still open issues. On the other hand, proprietary tools are reliable but rarely open-acess. All in all, most of the aforementioned implementations lack one or more of the following desirable properties:

1. **Radiometric Consistency**. The light interaction with the target and the sensor should be modeled according to physical-based reflection models whose parameters may vary across the spatial and spectral range. Although many open-source render engines are useful for generating photorealistic images, they may lack physical representativeness and/or modeling flexibility, which limits their degree of radiometric realism [14]. Moreover, these tools are often devoted to the human vision and they do not possess the photometric accuracy that space sensors are sensitive to [8]. Finally, an established pipeline for geometric and especially radiometric validation is rarely present.
2. **Physical Correspondence**. We define as *physical correspondence* the ability for a rendering pipeline to link physical quantities to each pixel in the image (e.g., photon flux, flux density) and to the global scene (e.g., exposure time). Even when the process is physically based, the output may lack this correspondence. For instance, Blender does not provide spectral and physical data along with the generated images, given that its rendering pipeline relies on path samples instead of actual photon fluxes [14]. In PANGU, there is no concept of exposure time [9]: "users simply set the relative brightness of light sources, and the resulting pixel levels are essentially a measure of the intensity of reflected light".
3. **Rendering Performance and Precision**. High frame-rate rendering is desirable to enable closed-loop V&V or to generate large datasets without prohibitive times. Unfortunately, the rendering task is typically computationally intensive as it requires the manipulation of large-size, high-resolution texture maps to reproduce the terrain's appearance and its interaction with light rays. This constrains the hardware resources, especially when moving from far to close ranges in a single simulation. In this case, a global map of the surface is required, and the Random Access Memory (RAM) may not be enough. For instance, Blender converts the entire displacement map into a 3D mesh, leading quickly to RAM saturation. Another major issue is the scale of the scene, which may cause precision losses due to large relative distances between the Sun and the target, placed at thousands to millions of kilometers, and the surface features, which have meter-scale variability. This issue implies that double-precision floating-point is desirable. To retain pixel-level accuracy and avoid aliasing, high-density sampling is typically employed, which involves a large number of independent concurrent evaluations. This is a task best handled by the GPU, which could be beneficial but may require special implementation rules, such as using single-precision. All in all, deploying dedicated memory-management and pruning techniques would retain feasible frame rates without renouncing to fidelity and precision, while considering the available hardware resources. However, this is not always possible with the black-box current available tools.
4. **Modeling Flexibility**. In many cases, the modeling of the sensor behavior in terms of distortion, diffraction, and noise cannot always be simulated by post-processing the final image. This often happens in black-box implementations where the image is the only available output. Instead, some optical effects or detector noises should be implemented directly inside the rendering procedure, before image generation. This consideration also applies to all those effects that are likely observable only in a space environment, such as the light-time travel effect.

Table 1
Currently available tools tailored to the rendering of celestial bodies.

	Technique	Core	License	Properties[a]
SurRender	Path-tracing or rasterization	OpenGL[b]	Commercial	1, 2, 3, 4
PANGU	Rasterization	OpenGL	Commercial	1, 3, 4
IRIS	Ray-tracing	OptiX	Restricted	1, 2, 3
NaRPA	Path-tracing	Mitsuba	Restricted	1, 2, 3
SISPO	Path-tracing or rasterization	Cycles or OpenGL	Open-source	1, 4
CORTO	Path-tracing	Cycles	Open-source	1, 3
GUISS	Path-tracing	Cycles	Open-source	1, 3
DUST	Hybrid[c]	Unreal Engine	Restricted	1, 3

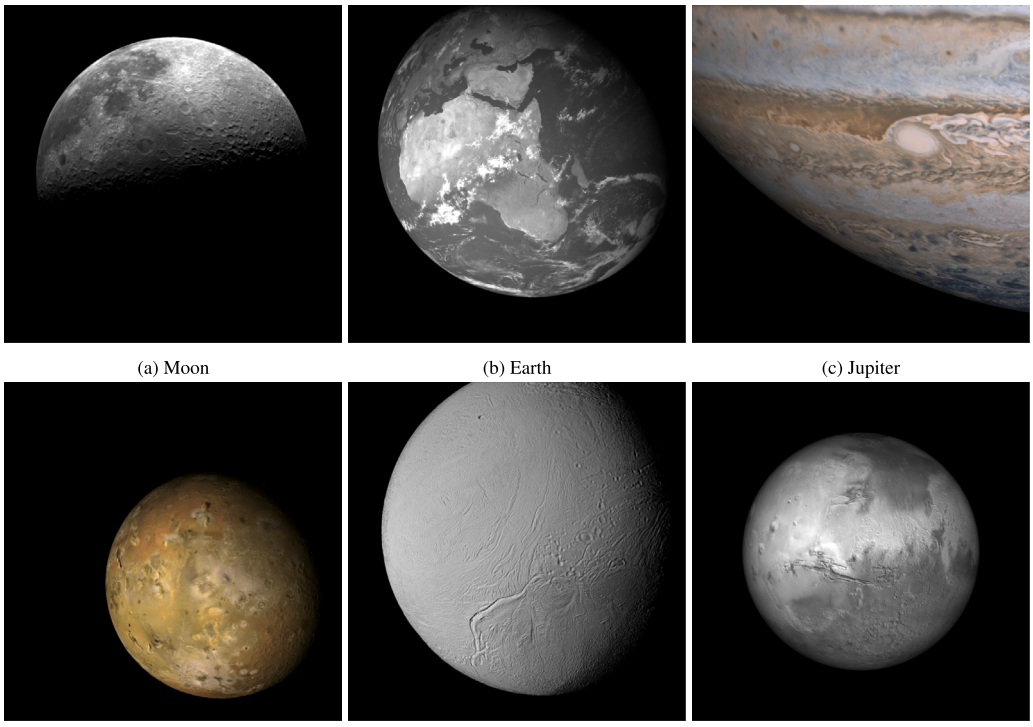
[a] To the authors' knowledge and experience with the tools.
[b] For rasterization.
[c] Blend of rasterization and path-tracing.

Table 1 lists all the tools available in the community, along with their main characteristics and whether they possess such properties.

A generalized tool that is capable of simulating any scene and at the same time complies with all these properties does not currently exist. Instead, it may be more convenient to focus on developing a tool tailored to specific scenarios, so as to relax some hard requirements that would be impossible to comply with in an all-purpose solution. With this in mind, this article proposes an efficient solution for spectral renderings of spherical and quasi-spherical bodies, including ellipsoids or irregular surface bodies, yet maintaining radiometric consistency and correspondence of physical quantities. The scenario of viability for such a solution is the imaging at far to mid-range distances, with a single light source and a single object in the scene. This is typically the case for the images taken by a probe visiting a celestial body.

The article will depict techniques based on smart bounding, sampling, and pruning of the discretization points, which enable relatively fast and precise computations. Moreover, the paper will show how different reflection models can be used and mixed together, as well as how the local radiometric and geometric parameters can be interpolated from texture maps. Finally, the methodology will be validated with an incremental validation pipeline to compare the synthetic imagery against analytical ground truth and real images. It is worth noting that

[6] https://www.mitsuba-renderer.org/.
[7] https://docs.blender.org/.
[8] https://www.unrealengine.com/.



(a) Moon (b) Earth (c) Jupiter

(d) Io (e) Enceladus (f) Mars

Fig. 1. Example images of planets and moons rendered with this methodology.

the validation is an important step often not disclosed in state-of-the-art works.

An implementation of the proposed methodology in MATLAB is left available as an open-access CPU-based application[9] for interested users and developers. The tool is flexible and easily adaptable to existing simulation architectures, thanks to an intuitive interface and simple configuration of input files. It is worth noting that the tool can also be used to cross-validate existing render engines, owing to its successful validation against real images.

To appeal to the reader's interest, we leave in Fig. 1 some examples of planets and moons across the solar system synthetically generated with this technique, before entering the technical discussion.

The article is structured as follows. A brief overview of the available rendering tools and methods within the space imaging field has been given in this chapter. The rendering problem is articulated mathematically in Section 2, while the key contributions of this work are highlighted in Section 3. A high-level solution to the problem is provided in the first part of Section 4, with implementation details and low-level techniques in the subsections that follow. Then, the methodology is thoroughly validated in Section 5. Finally, results, benchmarking, and applications are reported in Section 6 before the concluding remarks of Section 7.

2. Problem formulation

This article addresses a problem that intersects the fields of radiometry and computer graphics. This problem was first presented in the form of the *rendering equation* by Kajiya [37], which basically expresses the equilibrium of energy incident to and reflected from a surface. Since then, several rendering techniques were proposed that attempted to solve this equation.

In this section, a general formulation of the problem is first presented to compute the signal S in electrons per second collected by a camera that is observing a generic surface lit by a collimated light source. Then, the problem of merging all the contributions from multiple surfaces into a discrete – albeit coherent – pixel array is shown.

Table 2
Radiometry nomenclature.

Name	Symbol	Unit of measure
Radiance	L	$\mathrm{W\,sr^{-1}\,m^{-2}}$
Intensity	I	$\mathrm{W\,sr^{-1}}$
Exitance	M	$\mathrm{W\,m^{-2}}$
Flux density	F	$\mathrm{W\,m^{-2}}$
Irradiance	E	$\mathrm{W\,m^{-2}}$
Power	P	W
Projected solid angle	Ω	sr
Solid angle	ω	sr

The nomenclature used for the radiometry quantities follows the Nicodemus standard [38], with the only difference of using P as label for the power to distinct from Φ, which refers to the generic longitude coordinate. The nomenclature is summarized in Table 2.

The *radiance* is defined as the intensity per unit projected area [39]:

$$L = \frac{\mathrm{d}I}{\cos(\theta)\,\mathrm{d}A} \tag{1}$$

where θ is the angle of the light ray with respect to the normal of the surface A and I is the light intensity. Radiance is conserved for ideal optical systems.

M, F and E have the same unit of measure, but they express different quantities. The *exitance* is related to the light emitted by a surface and is obtained by integrating the radiance across the solid angle of emission. For a Lambertian point emitter, $M = \pi L$ applies as a result of spherical integration across a hemisphere [39]. Then, while the *flux density* is the power passing through a unit area, the *irradiance* is related to the light received by a surface with an arbitrary orientation. This implies that $E = F\cos\theta$. If the surface normal is parallel to the light direction, the irradiance is equal to the flux density.

[9] https://github.com/andrepiz/abram.

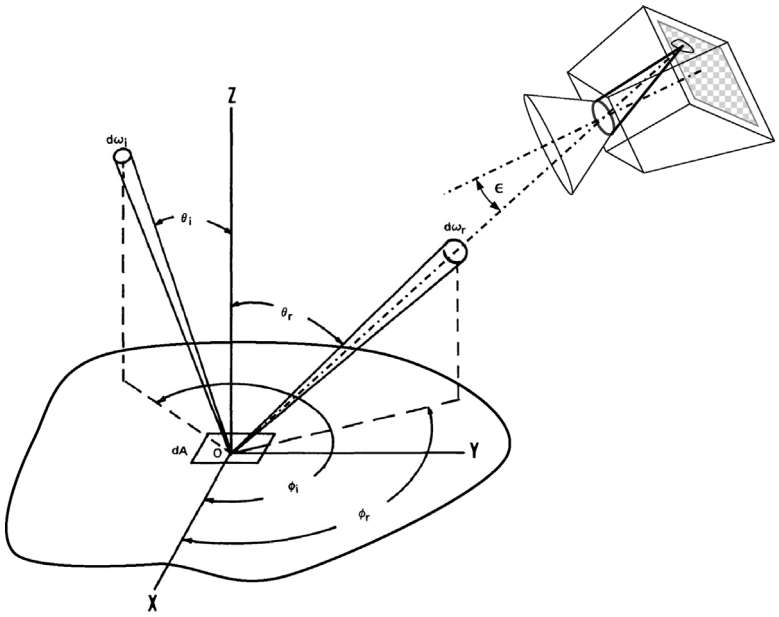


Fig. 2. Light interaction in Object-fixed frame (O).

All the defined quantities have spectral counterparts, meaning they can be defined per unit wavelength. The spectral quantity is labeled with a λ subscript. The bolometric quantity is obtained by integrating the spectral quantity over the desired bandwidth.

Now consider a camera observing a generic surface A illuminated by a collimated light source of spectral flux density F_λ, as shown in Fig. 2. The BRDF links the radiance reflected by an object with its irradiance. In the most general case, it is a function of wavelength, co-elevation (θ), and azimuth (ϕ) angles of incidence and reflection with respect to the normal:

$$f_r(\lambda, \theta_i, \theta_r, \phi_i, \phi_r) = \frac{L_r}{E} \tag{2}$$

Combining Eqs. (1) and (2), the spectral power P_λ collected by the optical head can be expressed as the integral of the reflected spectral intensity over the projected spherical angle Ω:

$$P_\lambda = \int_{\Omega_r} I_{r\lambda}\, \mathrm{d}\Omega = \Omega_r \int_A L_{r\lambda} \cos\theta_r \mathrm{d}A = \Omega_r \int_A f_r E_\lambda \cos\theta_r \mathrm{d}A$$
$$= \Omega_r F_\lambda \int_A f_r \cos\theta_i \cos\theta_r \mathrm{d}A \tag{3}$$

Namely, $\Omega_r = \omega_r \cos\epsilon$, being ϵ the off-point angle of the camera boresight with respect to the camera-to-object direction and ω_r the spherical angle in the reflection direction. For small and regular surfaces, the two can be assumed constant throughout A and taken out of the integral.

Eq. (3) has to be solved for each surface present in the scene. The more regular the surface, the easier the integral is to solve. It is worth noting that the surface can be a simple primitive (e.g., triangle, square) that is part of a larger mesh modeling an irregular body.

Finally, the signal collected by the camera, expressed in electrons per second, is given by

$$S = \int_{\Delta\lambda} \mathrm{QE}_\lambda \, \mathrm{T}_\lambda \, \frac{\lambda}{hc} \, P_\lambda \, \mathrm{d}\lambda \tag{4}$$

considering the lenses' Transmittance (T) and the detector's Quantum Efficiency (QE). While T reduces the number of photons arriving at the detector due to impurities and filters, QE converts photons into electrons at detector level with a given spectral efficiency. Note that the conversion factor $\frac{hc}{\lambda}$ (i.e., the power of a photon) is used to transform radiant power into photon flux before the integration. This step is necessary as the energy carried by a photon is a function of its wavelength.

Regardless of the number or shape of the surfaces, the problem of image formation arises. The camera frame is represented as a discrete pixel grid where each pixel collects the signal only from the part of the scene that it sees. This does not always (almost never) coincide with a uniform and easy-to-integrate surface. To tackle this problem, several computer graphics techniques have been proposed in the past years [40]. Two of the most popular are depicted in Fig. 3.

In *ray-tracing* [41], one or more rays are back-projected from each camera pixel, and their intersections with the bodies in the scene are computed. At each intersection point, a *shadow* ray is cast towards the known light emitters. If the light is hit, the point is valid, and the associated radiometric quantities can be computed. Usually, the BRDF function is simply evaluated at the point, instead of integrating it over the surface, so this method approximates the integral. The most basic and first version of ray-tracing was *ray-casting* [42], where the ray bounced only once back to the light source. A more advanced version is instead represented by *path-tracing* [43,44], where multiple rays are scattered back from the surface with a distribution that follows the one dictated by the BRDF, and they can bounce multiple times against other elements in the scene. This enables capturing soft shadows at the cost of lower frame rates. In fact, the computational time of ray-tracing techniques scales with the number of pixels but also with the number of bounces.

In *rasterization* [45], each primitive is projected onto the pixel array, and the energy carried by each primitive is spread over the pixels intersected by the projected shape. This allows computing the integral in Eq. (3) over the primitive shape instead of approximating at a single or multiple points as done in ray-tracing. This formulation scales with the number of primitives and is in general faster than ray tracing. The technique enables the rendering of occlusions using depth buffer maps [46], but it is complex to simulate the effects of multiple lights, reflections, and shadows.

In this article the imaging of planets and moons is considered. In most cases, these are quasi-spherical bodies mainly illuminated by a single source of light, placed at a far distance. Moreover, the BRDF that describes the reflection of celestial bodies is often approximated with simplified analytical models due to the lack of observational data. These considerations ease the resolution of the integral in Eq. (3). Thanks to spherical geometry considerations, a rendering technique can be devised to better manage computational resources while returning accurate renderings without any strong approximations.

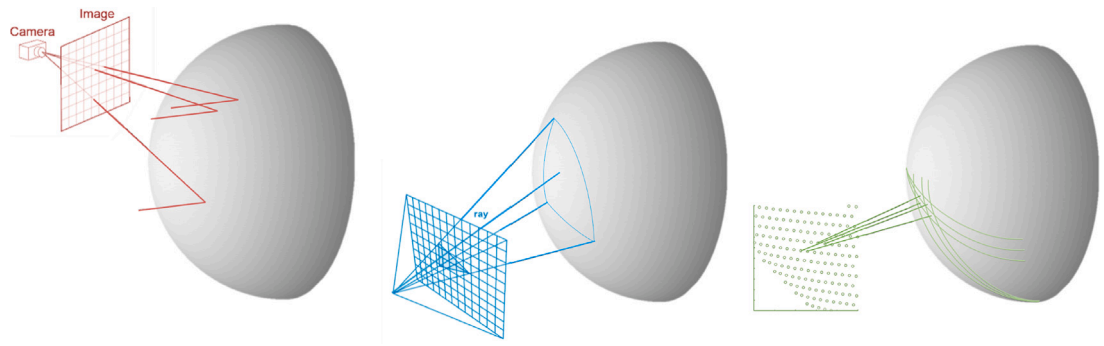


Fig. 3. Visualization of the ray-tracing (left) and rasterization (middle) concepts compared to our approach (right).

3. Contributions

This article brings three main contributions to the field of physically-based rendering applied to the space environment.

Physically-based rendering methodology. A novel rendering methodology is proposed that exploits the specific application's environment (quasi-spherical bodies, single light source, far distances, scarce data) to simplify the complexities involved in conventional rendering pipelines. Most importantly, it is ensured that radiometric consistency and physical correspondence are preserved by drafting the rendering equations from first principles of radiometry and ensuring energy conservation along the whole process. This is a fundamental requirement for carrying out camera design studies or using radiometrically-calibrated optical stimulators, as non-exhaustive examples.

Efficient computer graphics techniques. Smart sampling, geometry-aware pruning and energy-conserving gridding techniques enable computational efficiency and numerical accuracy while avoiding image artifacts. These features ensure the pipeline remains competitive with respect to currently available tools and can be run on common hardware resources.

Comprehensive multi-scale validation. A three-fold validation is conducted using phase laws, analytical models and real images, to validate the methodology at different metrics and scales, both globally (sphere-integrated) and locally (pixel-wise). This validation fills a common gap in similar works and can serve as a reference for users interested in validating their own rendering application.

4. Methods

This section begins with a high-level overview of the methodology, followed by detailed algorithms and techniques in subsequent subsections. Readers familiar with computer graphics problems may skim those parts and focus only on the first subsection. Nevertheless, since they also present contributions by proposing tailored solutions to these problems for the scenario of interest, readers new to the field or interested in implementation specifics are encouraged to explore the full details there.

4.1. Rendering methodology

Our approach lies in the middle of a rasterization and ray-tracing algorithms, and it is based on five main steps:

1. Discretization and sampling of the visible surface of the body with longitude–latitude *quadrangles*.
2. Pruning of non-contributing points through assessment of observability, illumination and occlusion conditions.
3. Generation of a 3D point cloud through integration of the radiometry in spherical coordinates.
4. Projection, distortion and direct gridding of the point cloud values into the discrete pixel array.
5. Application of noises and image generation.

Within the discussion of the procedure, the four frames depicted in Fig. 4 are used:

- **Body Fixed Frame (B)**. This frame has the origin at the center of the target body. The X_B axis passes through the equator, whereas the Z_B axis points to the north pole and the Y_B axis completes the frame. This frame is where texture maps of the bodies are defined: the point at zero longitude and latitude coordinates is the intersection of X_B with the equator.
- **Camera Frame (C)**. The origin of this frame is at the camera location. The Z_C axis is aligned with the camera boresight, while X_C and Y_C axes define the horizontal and vertical coordinates pointing respectively to the right and downward with respect to the pixel array.
- **Camera-Sun Frame (CSF)**. This frame is centered on the target body, with the X_{CSF} axis aligned with the Sun direction and the Z_{CSF} axis perpendicular to the plane identified by the Sun and camera directions. The Y_{CSF} axis completes the right-handed frame.
- **Object-Fixed Frame (O)**. This is a local frame placed on a generic point of the body surface, whose Z_O axis is aligned with the local normal, X_O points the local south direction, and Y_O completes the frame.

The source of light for the scenario of interest is the Sun, which can be assumed to be a point-wise Lambertian emitter [39], i.e. $M = \pi L_e$ with a black body spectrum. This implies that the spectral radiant flux density arriving at a point located at a distance d_{sun} is given by:

$$F_\lambda = \frac{4\pi R_{sun}^2}{4\pi d_{sun}^2} M_\lambda = \frac{A_{sun}}{d_{sun}^2} L_{e\lambda} \tag{5}$$

where A_{sun} is the frontal area of the sun and $L_{e\lambda}$ is the emitted spectral radiance of a black body as given by:

$$L_{e\lambda} = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_b T}} - 1} \tag{6}$$

By using the values for Earth and Sun (i.e., $T = 5782\,\mathrm{K}$, $d_{sun} = 1\,\mathrm{AU}$, and $R_{sun} = 6.95 \times 10^8\,\mathrm{m}$) and integrating over the entire spectrum, Eq. (5) returns the well-known solar flux density at Earth of about $F = 1373\,\mathrm{W\,m^{-2}}$.

The projected spherical angle that the pupil area of the camera A_{cam} subdues to a point placed at distance d_{cam} is given by [39]:

$$\Omega_r = \frac{A_{cam}\cos\epsilon}{d_{cam}^2} \tag{7}$$

At this point, Eqs. (5) and (7) are substituted into Eq. (3):

$$P_\lambda = L_{e\lambda} A_{sun} A_{cam} \int_A \frac{\cos\epsilon}{d_{sun}^2 d_{cam}^2} f_r(\lambda, \theta_i, \theta_r, \phi_i, \phi_r) \cos\theta_i \cos\theta_r \, \mathrm{d}A \tag{8}$$

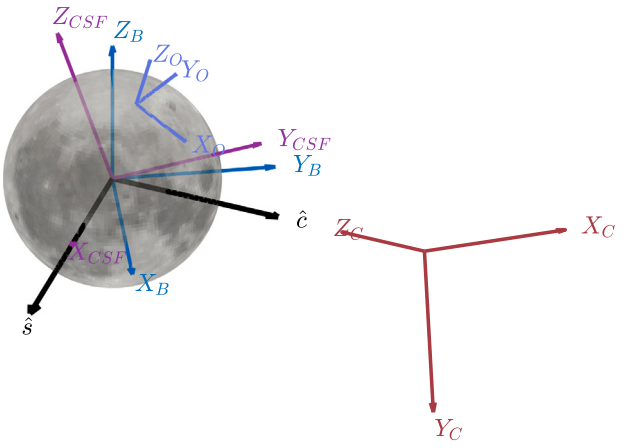


Fig. 4. Definition of the reference frames adopted in the discussion. The symbols $\hat{s}$ and $\hat{c}$ label the directions of the Sun and the camera, respectively. The B frame is defined by the body's longitude–latitude coordinate system, whereas the CSF frame depends on the relative camera-Sun geometry. The O frame represents a generic surface patch on the body and is aligned with the local surface normal. The frame C is aligned to the camera boresight and to the pixel array horizontal and vertical directions.

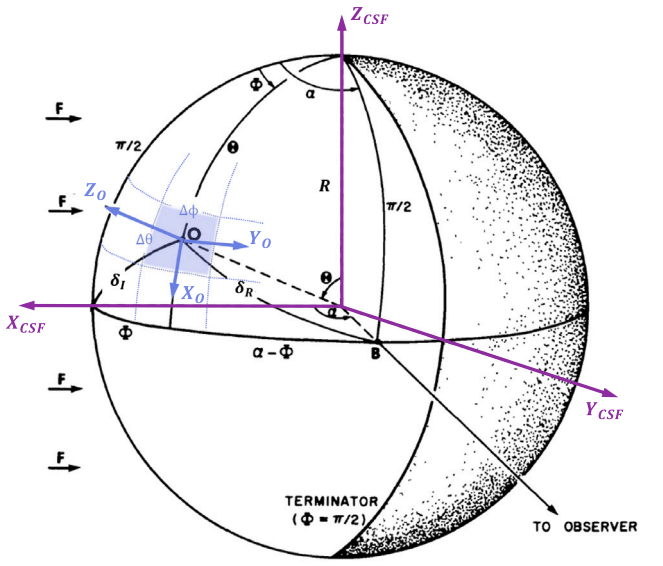


Fig. 5. Integration of radiometry in Camera Sun Frame. Instead of integrating across the entire illuminated sphere domain, in our approach the integration is split into smaller quadrangles defined by longitude and colatitude boundaries. *Source:* Picture modified from [52].

which can be integrated over the desired bandwidth $\Delta\lambda$:

$$P = \underbrace{A_{sun}\, A_{cam} \int_A \frac{\cos\epsilon}{d_{sun}^2 d_{cam}^2} f_r(\theta_i, \theta_r, \phi_i, \phi_r) \cos\theta_i \cos\theta_r \, \mathrm{d}A}_{(P/L)} \int_{\Delta\lambda} L_{e\lambda} \mathrm{d}\lambda \quad (9)$$

where the geometry-dependent contribution is separated from the spectral-dependent one by introducing the Power-to-Radiance (P/L). This coefficient only depends on the relative geometry between light, surface, and camera. It is measured in $\mathrm{sr\,m^2}$, and it can be seen as the fraction of emitted radiance that is converted to collected power. The separation of the (P/L) term assumes spectral independence of the BRDF across each waveband $\Delta\lambda$, which is a valid assumption considering that the reflection properties of celestial bodies are usually available at restricted wavelengths [47–50]. Even for well-known objects such as the Moon, where a standard model is available (see Sato et al. [51]), the maps of the spectral-dependent parameters are provided at separate bandwidths.

Substituting Eq. (8) into Eq. (4), the signal S in electrons per second follows a similar formulation:

$$S = (P/L) \int_{\Delta\lambda} \mathrm{QE}(\lambda)\, \mathrm{T}(\lambda)\, \frac{\lambda}{hc}\, L_{e\lambda} \, \mathrm{d}\lambda \quad (10)$$

The spectral integral that multiplies the (P/L) coefficient in Eqs. (9) and (10) is a scaling factor for the *total* amount of light the sensor perceives, regardless of the surface shape. It has spectral dependency, as for each waveband, the fraction of incoming light and collected light is different depending on the black body emission, camera detector, and optics response.

Let us first analyze how (P/L) can be computed for a sphere and assuming the Sun and the camera placed at far distances. Within these approximations, ϵ, d_{sun} and d_{cam} can be approximated constant and taken out of the integration. Considering a sphere, the surface integral can be rewritten as function of the phase angle α, parametrizing the BRDF by the longitude Φ and colatitude Θ angles referred to the CSF frame, as detailed in Lester et al. [52]. By definition of this frame, the terminator lies at $\Phi = \pi/2$, whilst the lit limb at $\Phi = \alpha - \pi/2$ (see Fig. 5). The colatitude spans 0 to π. Considering these integration limits, the (P/L) coefficient is computed as:

$$(P/L) = A_{sun}\, A_{cam} \frac{\cos\epsilon}{d_{sun}^2 d_{cam}^2} R^2 \underbrace{\int_{\alpha-\frac{\pi}{2}}^{\frac{\pi}{2}} \int_0^{\pi} f_r(\Phi, \Theta) \sin^3\Theta \cos(\Phi) \cos(\alpha - \Phi) d\Theta d\Phi}_{p_G \Psi(\alpha)} \quad (11)$$

Refer to Lester et al. [52] for the transformations involved in the integral. The so-called phase law $\Psi(\alpha)$, scaled by the geometric albedo p_G, condenses the full spherical integral into a simple scalar function and enables the computation of the integrated signal as if it was all collected by a single pixel.

This formulation lacks generality for extended bodies, i.e., those that project to more than 1 pixel in the image frame. In this case, the integral could be split for each surface $j = 1, \ldots, N$ belonging to the body. The main idea behind this article is to use discretized longitude–latitude sectors, also known as *quadrangles*, to evaluate the (P/L) contribution from each sector separately and fast, albeit ensuring radiometric fidelity:

$$(P/L)_j = A_{sun}\, A_{cam} \frac{\cos\epsilon_j}{d_{sun_j}^2 d_{cam_j}^2} \times R_j^2 \underbrace{\int_{\Delta\Phi_j} \int_{\Delta\Theta_j} f_{r_j}(\Phi, \Theta) \sin^3\Theta \cos(\Phi) \cos(\alpha - \Phi) d\Theta d\Phi}_{F_{r_j}(p_j, \alpha, \Phi_j, \Theta_j, \Delta\Phi_j, \Delta\Theta_j)} \quad (12)$$

where $\Delta\Phi_j$ and $\Delta\Theta_j$ are the quadrangle longitude and colatitude boundaries and F_{r_j} denotes the local BRDF integral.

Eq. (12) can be used to generate an array of values that express the fraction of total collected light for which each quadrangle on the sphere is responsible. The transformation of the integrated quadrangle values from the scene 3D domain to the image 2D domain is then accomplished using Direct Gridding, as it will be explained later.

The motivation behind such a discretized formulation is fourfold:

1. The aforementioned assumptions of far distances can be relaxed as the true values of ϵ, d_{cam}, and d_{sun} can be computed for each quadrangle. This allows for a correct rendering at closer range distances and off-nadir boresight pointing.
2. Point-wise values for the radius R_j, albedo p_j and incidence/reflection angles θ_{i_j} and θ_{r_j} can be used. These can be interpolated or computed from user-provided texture maps of displacement, albedo, and normal maps, respectively.
3. The CSF frame is ideal to avoid singularities and artifacts that may occur when imaging polar regions since the camera will

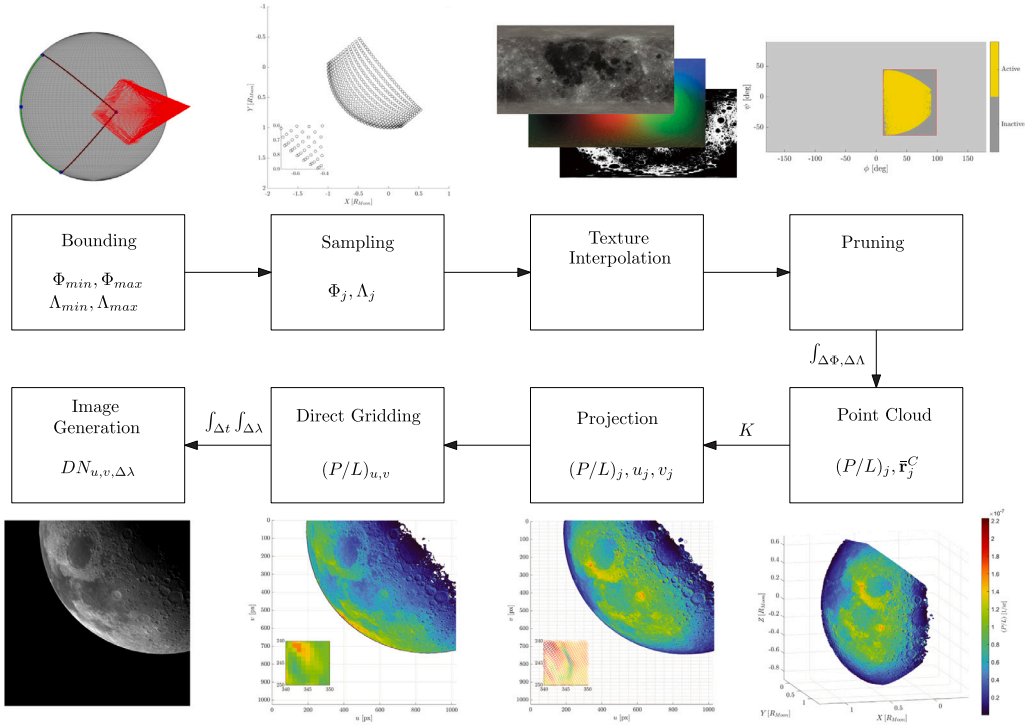


Fig. 6. Flowchart of the rendering methodology.

always lie in the XY plane. This also enables coherent sampling and bounding of the imaged region.

4. The spherical geometry favors the integration in contrast to more complex primitive shapes. In fact, for simple BRDFs an exact analytical solution for F_r exists. For more complex models, the integral can be solved numerically or relaxed by assuming constant values over the integration.

The finer the discretization, the more accurate the computation will be as the properties of each quadrangle, such as its albedo or its distance from the camera, can be assumed to be constant throughout the local integration. It is worth noting that even if this procedure requires the generation of a large number of discretization points, many of these can be pruned away before integration thanks to geometry and visibility reasons.

The overall procedure is shown in Fig. 6. Firstly, the sampling limits of the sphere are found, and the surface defined by such limits is discretized with $j = 1, \dots, N$ quadrangles defined by their longitude and latitude boundaries $\boldsymbol{\Phi}_j$ and $\boldsymbol{\Lambda}_j$. The local properties of each quadrangle, such as albedo, displacement, and normal are extracted and/or interpolated from textures when available. Then, points that do not respect visibility constraints are pruned away. At this stage, the array of $(P/L)_j$ values is computed by integrating Eq. (12) across each quadrangle. The quadrangle midpoint location $\bar{\boldsymbol{r}}_j^C$ is saved as well, obtaining a 3D point cloud, where the value of each point embeds the radiometric contribution of that quadrangle. Afterwards, the points are converted to image plane coordinates to apply distortions – if any – and then projected through the intrinsic camera matrix K to pixel coordinates u_j and v_j. The corresponding $(P/L)_j$ values are binned through Direct Gridding to obtain a 2D matrix $(P/L)_{u,v}$ where each element represents the value collected in a single pixel. The matrix is then converted with Eq. (10) to the signal S collected by each pixel at that waveband. Finally, noises and detector effects are applied, and the processed signal is integrated in time and quantized to obtain the image in Digital Number (DN).

For every step in the procedure, several methods and techniques have been developed to avoid generating artifacts in the final image and simultaneously reduce the computational burden. These will now be addressed separately in more detail.

4.2. Bounding of longitude and latitude domains

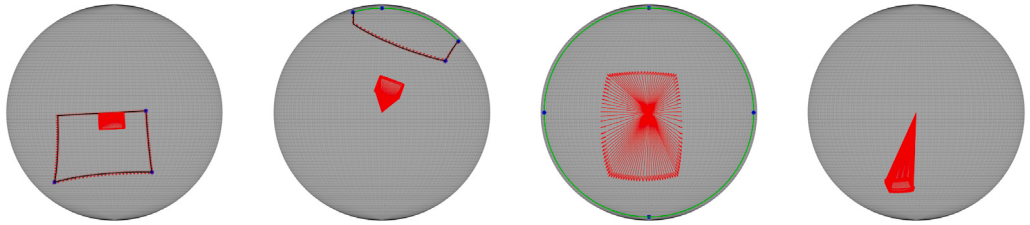
The bounding step is carried out to define the sampling limits, thus avoiding considering those quadrangles that fall outside the camera FOV. This is equivalent to the *frustum culling* functionality in common render engines. Pre-running this algorithm before the rendering greatly reduces the computational time as points are sampled only from a portion of the sphere, specifically within the $[\boldsymbol{\Phi}_{min}, \boldsymbol{\Phi}_{max}]$ longitude limits and the $[\boldsymbol{\Lambda}_{min}, \boldsymbol{\Lambda}_{max}]$ latitude limits.

To this aim, an efficient FOV intersection algorithm tailored for spheres was developed from scratch. In general, the FOV can be frustum-shaped or cone-shaped. The core of the algorithm is based on an iterative sampling of the FOV *perimeter* and an intersection check of each LoS belonging to the perimeter with a perfect sphere.

Firstly, a set of Lines Of Sight (LoSes) is sampled at coarse steps from the perimeter, and their intersections with the sphere are computed. At each iteration, the intersection points are transformed to longitude and latitude coordinates in the CSF frame, and their maximum values are saved. Then, in the following iteration, the number of points used to sample the FOV perimeter is increased. If the resulting longitude/latitude boundaries do not change over a certain tolerance, the process is stopped. Otherwise, another iteration is carried out.

Fig. 7 shows different solutions depending on the camera relative pose. The red points represent the intersections of the FOV perimeter, the blue points the computed longitude/latitude limits, while the green points the *tangency circle*. The tangency circle is defined as the locus of points that lie on the surface of the sphere whose points' directions towards the camera and towards the center of the sphere form an angle of 90 deg.

The solutions returned by the algorithm can be classified into three main cases:



(a) All intersecting (b) Missing intersections (c) No intersections, in FOV (d) No intersections, out FOV

Fig. 7. Intersection of a frustum-shaped FOV at different poses. The peripheral directions of the frustum may intersect the sphere in zero or multiple points. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

1. All the LoSes intersect the sphere, as in Fig. 7(a). This means that the background is not imaged by the camera, and boundaries are well defined. This is often the case for nadir-pointing close-range scenarios.
2. One or more LoSes are not intersecting the sphere, as in Fig. 7(b). This means that part of the background is imaged by the camera, such as during mid-range and off-nadir close-range scenarios. Nevertheless, to obtain meaningful boundaries, the missing intersection points are substituted with the points that belong to the tangency circle.
3. No LoSes intersect the sphere. This means that either the entire target is contained in the FOV, as in Fig. 7(c), or the target is not visible at all, as in Fig. 7(d). This often occurs in far-to-mid range scenarios. To distinguish between the two cases, the following is checked: if the maximum angle spanned by the LoS projected on a common plane is lower than the angle to the body limb, it means the target is completely outside. To clarify this, refer to Fig. 8.
   The normal to the plane identified by the boresight direction and the camera-to-target direction is found:

$$\hat{\boldsymbol{n}} = \hat{\boldsymbol{b}} \wedge -\hat{\boldsymbol{c}} \tag{13}$$

   Then, each LoS direction $\hat{\boldsymbol{d}}$ is projected onto this plane:

$$\boldsymbol{d}_{proj} = \hat{\boldsymbol{d}} - \left(\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{d}}\right) \hat{\boldsymbol{n}} \tag{14}$$

   The angles of these vectors with respect to the camera boresight $\hat{\boldsymbol{b}}$ are evaluated:

$$\delta_{LOS} = \arccos(\hat{\boldsymbol{b}} \cdot \hat{\boldsymbol{d}}_{proj}) \tag{15}$$

   as well as the angle of the body direction $(-\hat{c})$ with respect to the boresight:

$$\delta_{body} = \arccos(\hat{\boldsymbol{b}} \cdot -\hat{\boldsymbol{c}}) \tag{16}$$

   The half angular size of the target is:

$$\xi = \arcsin\left(\frac{R}{d_{cam}}\right) \tag{17}$$

   Hence, the limb angle is equal to:

$$\delta_{limb} = \delta_{body} - \xi \tag{18}$$

   To have the body completely outside the FOV, the maximum angle spanned by the LoSes must be smaller than limb angle:

$$\max\left(\delta_{LOS}\right) < \delta_{limb} \tag{19}$$

   If this condition is true, the rendering returns a black image as no points belonging to the body are visible. Otherwise, the tangency angles are used as longitude/latitude boundaries.

The bounding algorithm fails to return a meaningful solution when the observer is positioned away from the XY plane — for example, when looking from above at the polar regions of the sphere. However, this is not an issue since the camera will always lie in the XY plane per construction, as the CSF frame is used. Therefore, the limits will always be found regardless of the camera's pose in the actual body-fixed frame B.

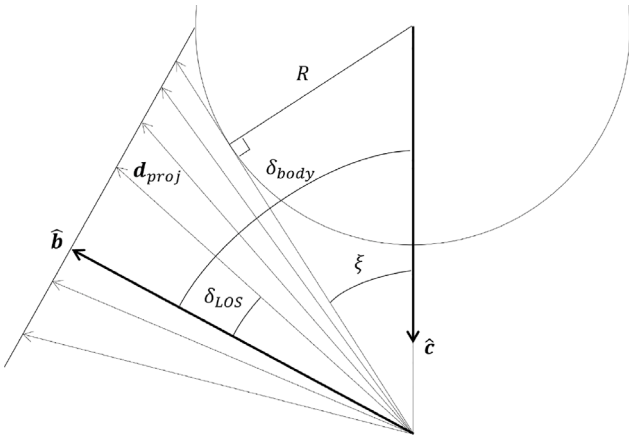


Fig. 8. When no LoSes intersect the sphere, the FOV intersection algorithm compares the body angular size ξ, the angle δ_{body} from the boresight to the center of the body and the angle δ_{LOS} spanned by each projected LoS.

4.3. Sampling of quadrangles

Sampling is necessary to define the position of each quadrangle in the scene and its integration limits. Sampling is done in the CSF frame in the longitude and latitude domains separately.

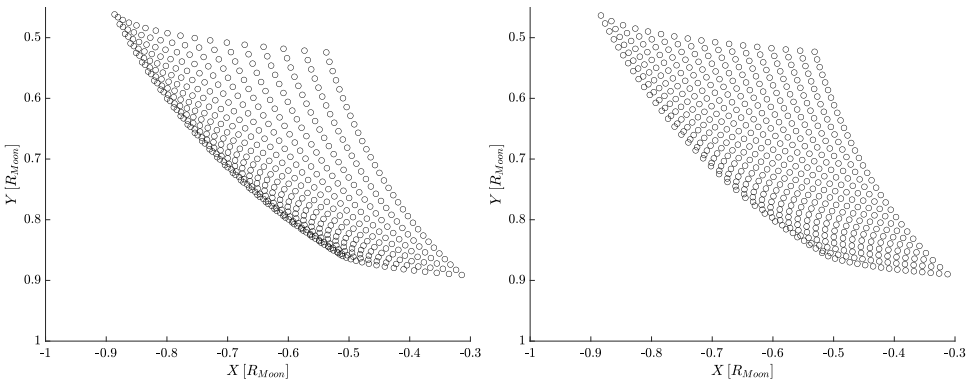
An ad-hoc *projected-uniform* sampling technique has been developed to address the issue of classical uniform sampling, where points after projection are concentrated much more at the limbs. The idea is to have a sampling of points such that their projection on a plane perpendicular to the camera-to-target direction is spread uniformly.

A comparison of a classical uniform sampling against the projected-uniform sampling is depicted in Fig. 9 for an off-nadir close-range pose, showing a more homogeneous distribution of points for the latter method.

This kind of sampling enhances the fidelity in those regions that occupy a larger space in the pixel array, but using the same number of points as a classical uniform sampling. It is also beneficial in diminishing the artifacts and improve the frame rate, especially in those scenarios that are computationally expensive due to the large number of points to process.

To understand how projected-uniform sampling is accomplished, consider Fig. 10. A view of the XY plane of the CSF frame is shown, with the Z axis exiting the page.

Considering a total of N_Φ longitude points, the goal is to sample Φ_k for $k = 1, \ldots, N_\Phi$ so that the interval projected by $[\Phi_k, \Phi_{k+1}]$ is constant.



(a) Uniform sampling (b) Projected-uniform sampling

Fig. 9. A projected-uniform sampling ensures a more homogeneous distribution of points on the pixel array.

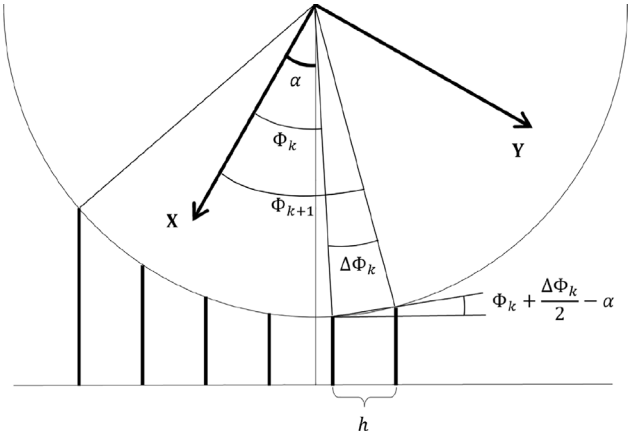


Fig. 10. Projected-uniform sampling iteratively finds the angle step $\Delta\Phi_k$ such that the projection of the arc spanned by that angle is a constant value h.

From trigonometric considerations, the interval h is given by:

$$h = 2 \sin\left(\frac{\Delta\Phi_k}{2}\right) \cos\left(\Phi_k + \frac{\Delta\Phi_k}{2} - \alpha\right) \tag{20}$$

The full span $H = h\, N_\Phi$ can be computed by plugging the known limits $[\Phi_{min}, \Phi_{max}]$ in the same formula:

$$H = 2 \sin\left(\frac{\Phi_{max} - \Phi_{min}}{2}\right) \cos\left(\Phi_{min} + \frac{\Phi_{max} - \Phi_{min}}{2} - \alpha\right) \tag{21}$$

Sampling is then carried out by solving iteratively the implicit equation:

$$2 \sin\left(\frac{\Delta\Phi_k}{2}\right) \cos\left(\Phi_k + \frac{\Delta\Phi_k}{2} - \alpha\right) - \frac{H}{N_\Phi} = 0 \tag{22}$$

starting from $\Phi_k = \Phi_{min}$. At each step, the angle step $\Delta\Phi_k$ and the sample $\Phi_{k+1} = \Phi_k + \Delta\Phi_k$ for the following iteration are found, for each point k up to N_Φ. The same scheme can also be used to sample the latitude points by setting the phase angle to zero.

Considering a total number of quadrangles N, the number of samples in the longitude and latitude domains are given by:

$$\begin{cases} N_\Lambda &= \sqrt{\frac{N}{\frac{1+\cos\alpha}{2}}} \\ N_\Phi &= N_\Lambda \frac{1+\cos\alpha}{2} \end{cases} \tag{23}$$

The two numbers are different due to the phase angle, which reduces the number of longitude samples with respect to the latitude ones. The coefficient used to scale down the number of samples is the fraction of illuminated disk area, which is given by $\frac{1+\cos(\alpha)}{2}$. Note that the latitude domain limits are not affected by the phase angle since the sampling is done in the CSF frame.

After sampling longitude and latitude points separately, the coordinates of each quadrangle are obtained by creating a mesh grid from the two coordinate vectors, then turning the grid into a vector. An array of longitude/latitude pairs $(\Phi, \Lambda)_j$ is obtained, which fully defines the boundaries of each quadrangle, given by:

$$Q_j = \begin{bmatrix} (\Phi_j, \Lambda_j) \\ (\Phi_{j+1}, \Lambda_j) \\ (\Phi_{j+1}, \Lambda_{j+1}) \\ (\Phi_j, \Lambda_{j+1}) \end{bmatrix} \tag{24}$$

The properties of the quadrangle, such as its position, normal and directions with respect to sun and camera have to be evaluated at a single point. The quadrangle midpoint is chosen as reference coordinate point:

$$(\bar{\Phi}, \bar{\Lambda})_j = \left(\frac{\Phi_j + \Phi_{j+1}}{2}, \frac{\Lambda_j + \Lambda_{j+1}}{2}\right) \quad \text{for} \quad j = 1, \ldots, N \tag{25}$$

4.4. Texture interpolation

As mentioned earlier, texture maps can be included in the rendering, when available, to increase the fidelity.

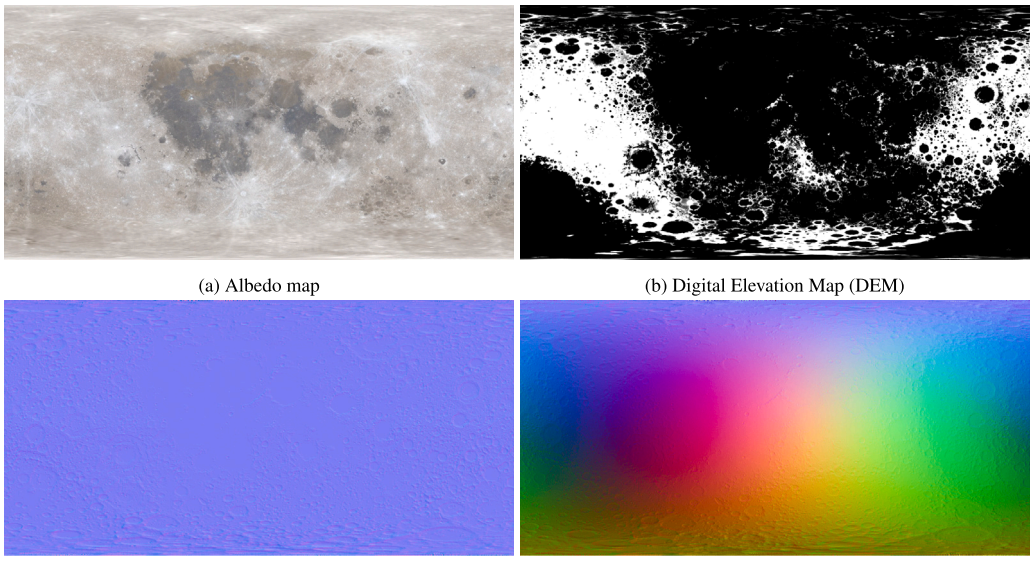
These maps are usually defined in the longitude and latitude coordinates of the body-fixed frame B, so they must be interpolated after rotating the coordinates originally sampled in CSF. In general, by calling $\mathcal{F}$ the gridded 2D interpolant function of a texture map, the following steps can be used to retrieve the texture value:

1. Convert the quadrangle location in CSF from spherical coordinates $[R, \bar{\Phi}_j, \bar{\Lambda}_j]$ to cartesian coordinates $\boldsymbol{r}_j^{CSF}$
2. Rotate the vector in the B frame using the Direction Cosine Matrix (DCM) $\boldsymbol{r}_j^B = A_{CSF \to B}\, \boldsymbol{r}_j^{CSF}$
3. Convert the vector $\boldsymbol{r}_j^B$ back to spherical coordinates $[R, \bar{\phi}_j, \bar{\lambda}_j]$
4. Evaluate the gridded interpolant at the corresponding longitude and latitude coordinates $(\cdot)_j = \mathcal{F}(\bar{\phi}_j, \bar{\lambda}_j)$

Note that in this way any region of the planet can be rendered, setting the correct orientation between the two frames. For instance, the south pole can be rendered by placing the observer at $\alpha = 90\,\text{deg}$ and rotating the B frame so that that region coincides with the CSF horizon, as shown with some rendering examples later in Section 6.

Three types of textures are considered, each one with different dependencies on the output:

- **Albedo**. The albedo map returns the local albedo $p_j = \mathcal{F}_p(\bar{\phi}_j, \bar{\lambda}_j)$ at each point. An example of rendering considering only the



(a) Albedo map (b) Digital Elevation Map (DEM)

(c) Normal Local Map (NLM) (d) Normal Body Map (NBM)

Fig. 11. Examples of texture maps of the Moon. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

albedo map is provided in Fig. 12(a). The local albedo only affects the radiometry as it changes the values of each $(P/L)_j$ through f_{r_j}.

- **Displacement**. The Digital Elevation Map (DEM) returns the local displacement $D_j = \mathcal{F}_D(\bar{\phi}_j, \bar{\lambda}_j)$ at each point, which is summed to the mean planet radius to obtain the local radius $R_j = R + D_j$. An example of rendering considering albedo and displacement maps is provided in Fig. 12(b). The local radius affects both geometry and radiometry as it changes the values of each $(P/L)_j$ and each $\boldsymbol{r}_j$.
- **Normal**. The normal map returns the local normal $\hat{\boldsymbol{n}}_j = \mathcal{F}_{\hat{\boldsymbol{n}}}(\bar{\phi}_j, \bar{\lambda}_j)$ at each point, which in turn changes the reflection and incidence angles θ_r and θ_i. Since it outputs a vector, it also has to carry the information on its frame of definition. We then define a Normal Local Map (NLM) when the frame is the local frame O and a Normal Body Map (NBM) when the frame is the body-fixed B. To visualize the normal map, each component of the normal vector can be encoded in the three channels of a RGB picture. For a NLM, the texture resembles Fig. 11(c), where the color is predominant on the blue channel as most of the normals are closely aligned to the upward direction Z of the O frame. On the other hand, the NBM has a rainbow pattern along the horizon as the direction of the normal rotates around the equatorial plane. The Moon NBM is shown in Fig. 11(d). At the north pole, the O and B frames are aligned and so the color as well. On the problem at hand, it is more efficient to provide the normal map already expressed in B frame to skip the frame transformation from the O frame to the B one. An example of rendering considering albedo, displacement and normal maps is provided in Fig. 12(c). The local normal only affects the radiometry as it changes the values of each $(P/L)_j$ through F_{r_j}.

Albedo and displacement texture maps are often released open-source by scientific teams and can be retrieved from different sources. Normal maps are more difficult to find, despite their importance in increasing the fidelity of the shadows. As a by-product of the proposed methodology, several normal maps were obtained [53] by processing the displacement maps with an algorithm that computed the mean of the normals of neighboring triangle facets for each pixel.

The closer the range, the higher the resolution of the texture map that is required to capture small-scale features and avoid the appearance of artifacts correlated to the discrete nature of the texture grid itself. However, this requirement puts some constraints on the hardware to dedicate for the rendering, as with high-resolution maps the memory gets filled up quickly. Future works involve the development of map cropping or clip-mapping techniques to allow close-range renderings on common hardware resources [54–56].

4.5. Pruning of inactive points

Before proceeding with the integration, it is convenient to prune all the points that do not contribute to the energy content of the scene. To this aim, the following conditions are evaluated:

1. **FOV**. Only those points that fall inside the FOV are retained, i.e., the bearing angle of each point has to be lower than the half-FOV. This is expressed by:

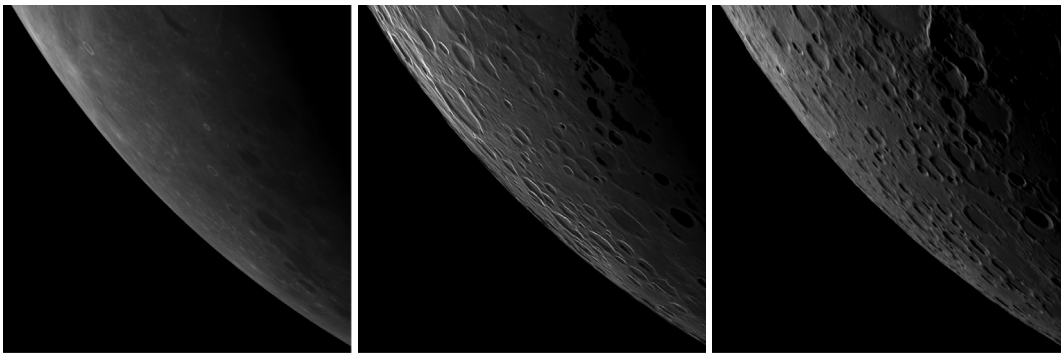
$$\beta_j = \arccos(\hat{r}^C_{z_j}) \leq \frac{\text{FOV}}{2} \tag{26}$$

for a cone-shaped FOV, or by:

$$\beta_{u_j} = \arctan\left(\frac{\hat{r}^C_{x_j}}{\hat{r}^C_{z_j}}\right) \leq \frac{\text{FOV}_u}{2} \quad \wedge \quad \beta_{v_j} = \arctan\left(\frac{\hat{r}^C_{y_j}}{\hat{r}^C_{z_j}}\right) \leq \frac{\text{FOV}_v}{2} \tag{27}$$

for a frustum-shaped FOV, where $\hat{\boldsymbol{r}}^C_j = A_{B\to C}\,\hat{\boldsymbol{r}}^B_j$, and FOV_u and FOV_v are the angular sizes of the FOV in the horizontal and vertical directions respectively.
Note that this step is not equivalent to the bounding explained in Section 4.2, which only provided the longitude and latitude sampling limits. Points inside these limits may still be outside the FOV.
2. **Observability**. For the point to be observable, the reflection angle, i.e., the angle between the local normal $\hat{\boldsymbol{n}}$ and the camera direction, must be lower than 90 deg. Namely:

$$\theta_{r_j} = \arccos(\hat{\boldsymbol{n}}_j \cdot \hat{\boldsymbol{c}}_j) < 90\,\text{deg} \tag{28}$$



(a) Albedo (b) Albedo and Displacement (c) Albedo, Displacement and Normal

Fig. 12. Moon limb with increasing level of fidelity.

where the local camera direction is given by:

$$\hat{\boldsymbol{c}}_j = \frac{\boldsymbol{r}_c - \boldsymbol{r}_j}{\|\boldsymbol{r}_c - \boldsymbol{r}_j\|} \tag{29}$$

where $\boldsymbol{r}_c$ and $\boldsymbol{r}_j$ are the camera and quadrangle position respectively.

3. **Illumination**. For the point to be lit, the incidence angle, i.e., the angle between the local normal $\hat{\boldsymbol{n}}$ and the Sun direction, must be lower than 90 deg. Namely:

$$\theta_{i_j} = \arccos(\hat{\boldsymbol{n}}_j \cdot \hat{\boldsymbol{s}}_j) < 90\,\text{deg} \tag{30}$$

where the local Sun direction is given by:

$$\hat{\boldsymbol{s}}_j = \frac{\boldsymbol{r}_s - \boldsymbol{r}_j}{\|\boldsymbol{r}_s - \boldsymbol{r}_j\|} \tag{31}$$

where $\boldsymbol{r}_s$ and $\boldsymbol{r}_j$ are the Sun and quadrangle position respectively.

4. **Occlusion**. When the displacement map is used, some points may not contribute to the image content due to *self-shadowing* caused by local surface morphology (e.g., craters, elevation changes, ...). To check this condition, a ray intersection algorithm is proposed as shown in Fig. 13(a).
The main idea is to check that the height of the ray H_{jk} from the sphere surface is always positive along the ray path direction $\hat{\boldsymbol{d}}_j$. For every point $k = 1, \dots, N_p$ sampled along the ray originating at the quadrangle location $\boldsymbol{r}_j$ with direction $\hat{\boldsymbol{d}}_j$, the following condition must be verified:

$$\begin{aligned} &\boldsymbol{r}^B_{jk} = \boldsymbol{r}^B_j + \hat{\boldsymbol{d}}^B_j\, d_k \quad \xrightarrow{car2sph} \quad \begin{bmatrix} r_{jk} \\ \phi_{jk} \\ \lambda_{jk} \end{bmatrix}^B \\ &\rightarrow \quad R_{jk} = R + \mathcal{F}_D(\phi_{jk}, \lambda_{jk}) \\ &\rightarrow \quad H_{jk} = r_{jk} - R_{jk} > 0 \end{aligned} \tag{32}$$

where $\boldsymbol{r}_{jk}$ is the position of the ray sample k that started at the quadrangle j and d_k is the ray sample distance. Note that the ray distances d_k can be sampled uniformly, log-spaced, or with ray-marching techniques [57].
A point can be occluded because its rays cannot reach either the Sun or the camera. For this reason, the occlusion check has to be done twice, setting the ray path first directed towards the Sun and then towards the camera. Thanks to the sphere curvature, it is possible to restrict the occlusion check only to those quadrangles that belong to the terminator and limb regions, provided there are no too large displacement shifts. The longitude extensions Φ_{span} of these regions can be found by considering the worst-case scenario of having the deepest point exactly at the terminator/limb region and the highest point at the last intersection of the ray, as depicted in Fig. 13(b):

$$\Phi_{span} = \arccos\left(\frac{R + \min(\mathcal{F}_D)}{R + \max(\mathcal{F}_D)}\right) \tag{33}$$

The occlusion check is therefore performed using the Sun direction for all the quadrangles that belong to the terminator region, i.e. whose longitude in the CSF frame is between $\pi/2 - \Phi_{span}$ and $\pi/2 + \Phi_{span}$, and using the camera direction for all the quadrangles that belong to the limb region, i.e. between $\alpha - \pi/2 - \Phi_{span}$ and $\alpha - \pi/2 + \Phi_{span}$.

For better clarity, the boolean masks corresponding to each one of the afore-mentioned conditions are depicted in Fig. 14, together with the overall mask of the active points obtained as the intersection of these boolean masks. The masks cover the sampled space of longitude and latitude coordinates: points outside the red box have been excluded from the sampling because of the initial bounding algorithm.

4.6. Integration and point cloud generation

Once all the energy-contributing quadrangles have been identified, the integration of the corresponding BRDF is carried out to obtain the 3D point cloud of $(P/L)_j$ values. This step is similar to the task performed by *fragment shaders* in modern rendering pipelines [58]: determining the final color of a fragment based on material properties, light interaction, and viewing direction. While in ray-tracing the fragment is the pixel itself, in this case the fragment is a quadrangle condensed into a point of the cloud.

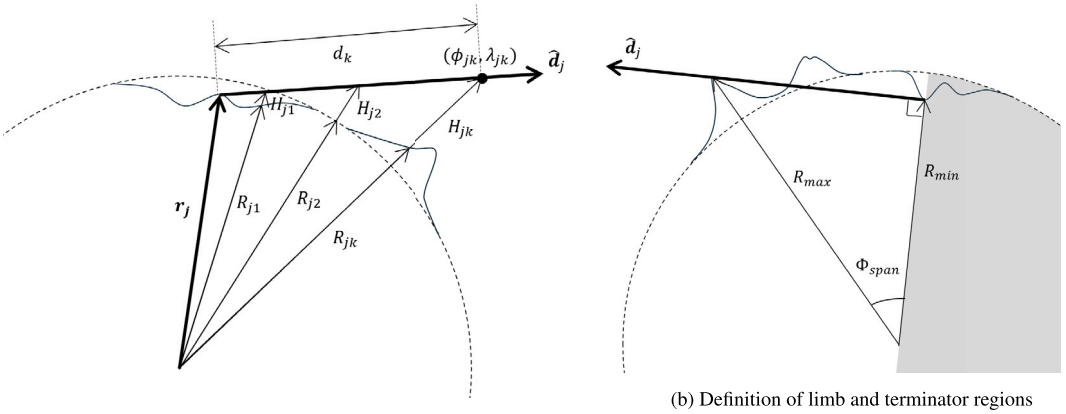
The BRDF integral term F_r in Eq. (12) can be computed with three methods of integration:

1. With the **integral** method, the following equation is solved:

$$F_{r_j} = \int_{\Phi_j}^{\Phi_j + \Delta\Phi_j} \int_{\Theta_j}^{\Theta_j + \Delta\Theta_j} f_{r_j}(\Phi, \Theta) \sin^3\Theta \cos(\Phi) \cos(\alpha - \Phi) d\Theta d\Phi \tag{34}$$

either exactly, when the analytical solution for the integral is available, or otherwise performing adaptive quadrature integration. Note that the latitude coordinate Λ where the quadrangles are defined must be converted to colatitude Θ before integration. As an example for an exact integral solution, the Lambert reflection model, whose BRDF is given by $f_r = p_N/\pi$, allows to solve the integral exactly and write it as a function of its integration limits:

$$F_{r_j} = \frac{p_N}{\pi} \frac{1}{48} \left(g(\Theta_j) - g(\Theta_j + \Delta\Theta_j)\right) \left(m(\Phi_j + \Delta\Phi_j, \alpha) - m(\Phi_j, \alpha)\right) \tag{35}$$



(a) Intersection algorithm

(b) Definition of limb and terminator regions

Fig. 13. The occlusion algorithm verifies whether a ray sampled from the quadrangle can reach the camera and the light by evaluating the local elevation along its path at different sample distances. The occlusion check is performed only on quadrangles belonging to the limb and terminator regions.

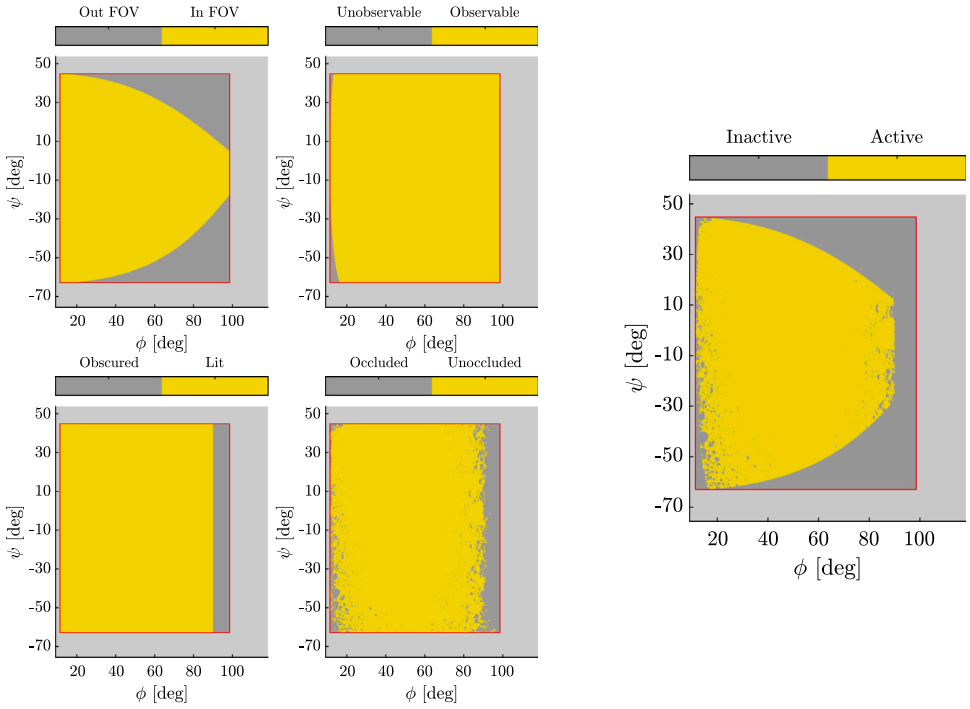


Fig. 14. Pruning of the inactive points on a sphere. The sampled domain is represented by the red longitude–latitude rectangle. Points that satisfy the condition are colored in yellow. A point is considered active, and therefore survives the pruning, if it satisfies all four conditions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where:

$$g(\Theta) = \cos(3\,\Theta) - 9\cos(\Theta)$$
$$m(\Phi, \alpha) = \sin(\alpha - 2\Phi) - 2\Phi\cos(\alpha) \tag{36}$$

and p_N is the Normal albedo.

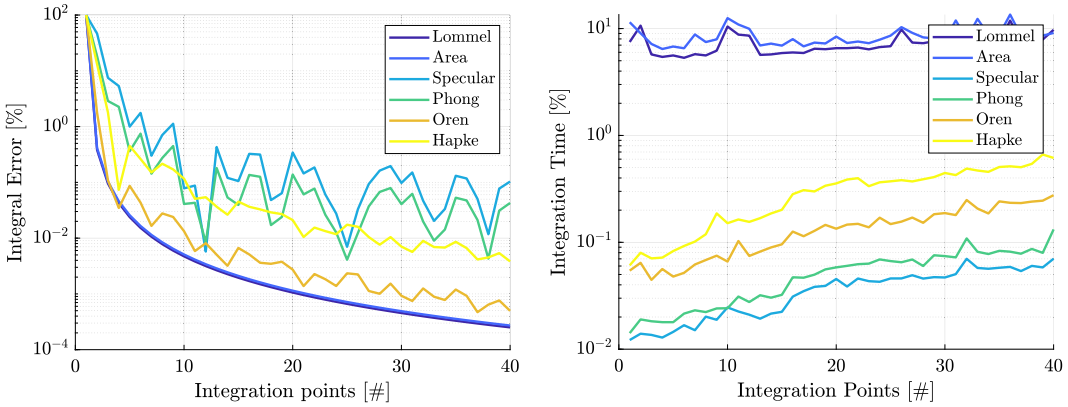
2. The **trapezoidal** method performs a numerical integration via the trapezoidal rule across the quadrangle boundaries at the specified integration points. The number of integration points can be chosen adaptively depending on the angle spanned by the quadrangle. This method is a trade-off between accuracy and rendering time, as shown in Fig. 15. The integration error and the integration time reduction with respect to the integral method are reported for different BRDF considering a quadrangle spanning 2 degrees with an increasing number of integration points. An empirically derived rule of thumb is to set at least 5 integration points per degree spanned along the sphere surface. Following this rule, we obtain integration times between 0.02% and 0.1% with respect to the integral method for the most complex BRDF, with only a decrease in accuracy lower than 0.1%.
3. The **pointwise** method evaluates the BRDF at the quadrangle midpoint – similarly to a ray-tracer – and multiplies the result by the unit-less quadrangle area to approximate the integral:

$$F_{r_j} = f_{r_j}(\theta_{i_j}, \theta_{r_j}, \phi_{i_j}, \phi_{r_j})\cos\theta_{i_j}\cos\theta_{r_j}\hat{A}_j \tag{37}$$

where:

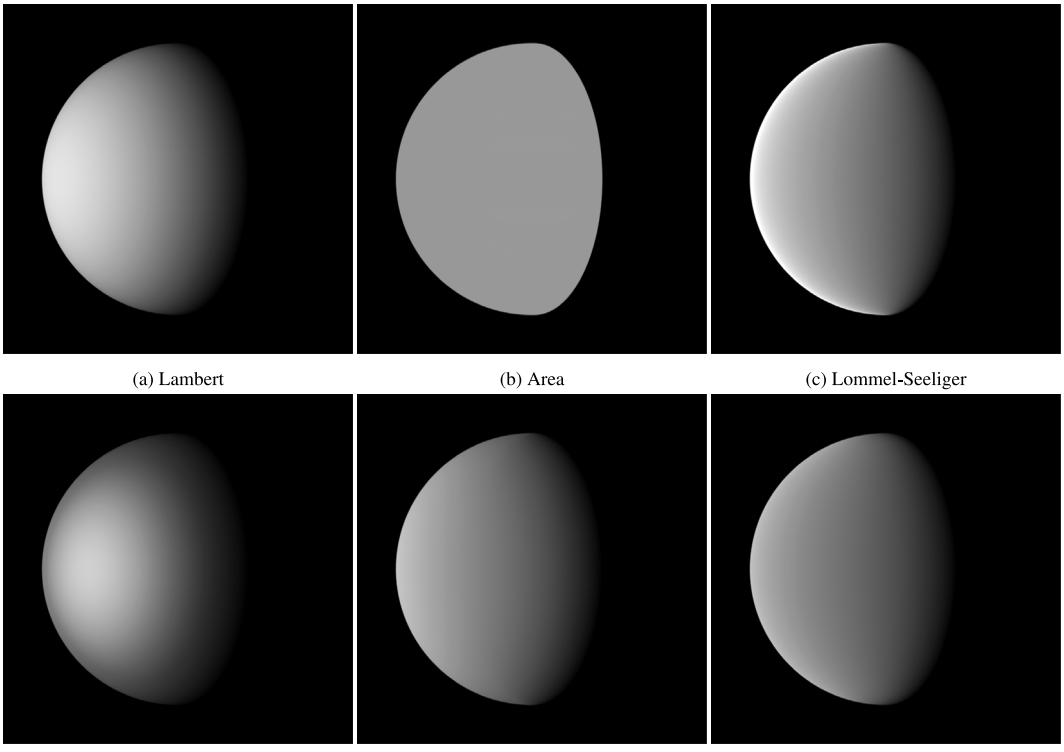
$$\hat{A}_j = \Delta\Phi_j\left(\sin(\Lambda_j + \Delta\Lambda_j) - \sin(\Lambda_j)\right) \tag{38}$$

If the points are sampled very fine, the point-wise integration accuracy increases while maintaining better performance than the previous methods. Therefore, when the random-access memory is large enough to store all the data points, this method



(a) Difference in the integrated value (b) Integration time reduction

Fig. 15. Evaluation of accuracy and performance of the trapezoidal method with respect to the integral method for different reflection models. Accuracy is quantified as the absolute difference between the integral solved via the trapezoidal rule and the one solved analytically or with adaptive quadrature. Performance is represented by the lower computation time that is obtained when using this approximation.



(a) Lambert (b) Area (c) Lommel-Seeliger

(d) Phong (e) Oren (f) Hapke

Fig. 16. Comparison of different BRDF for a sphere with constant albedo and phase angle of 60 deg. The parameters used in the Phong, Oren, and Hapke models were chosen arbitrarily.

is preferable due to its speed, especially when using complex BRDF.

The BRDF function and its parameters are inherently linked with the surface properties of the body and how it interacts with light. Consequently, different BRDF models shall be used depending on the body that is observed. Rendering examples of different BRDFs for a texture-less sphere are shown in Fig. 16 for the sake of completeness.

The Lambert model assumes light reflected isotropically, so that it depends only on the incidence angle. Conversely, the energy reflected with the Area model is independent of both the incidence and reflection directions. A Lambert model is commonly used as a first approximation to model the reflection of planets with an atmosphere [59,60]. For airless bodies with dark and diffuse surface such as asteroids, the Lommel-Seeliger model better approximates the regolith behavior [61]. Unfortunately, it predicts a strong opposition effect, and it is based on single scattering only. To overcome this limitation, the Hapke model was introduced [62,63] and it has become the standard model of reflection for the Moon and small bodies in the past years. Other reflection models exist, such as the Oren-Nayar [64], which models reflection of rough surfaces, and the Specular model, which is exploited

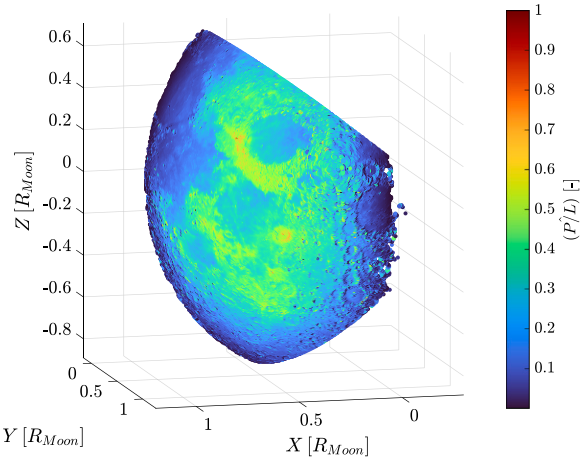


Fig. 17. 3D point cloud of (P/L) values normalized to the maximum. Each point of the cloud represents a quadrangle, and its value is the fraction of light emitted by the Sun that gets collected by the camera.

to render shiny surfaces. Finally, the Phong model is obtained as a weighted combination of Specular and Lambert models.

After each quadrangle is integrated, its $(P/L)_j$ coefficient is computed using Eq. (12). At this point, the 3D point cloud of quadrangle midpoint locations $\boldsymbol{r}_j^C$ and corresponding $(P/L)_j$ values, shown in Fig. 17, can be projected and gridded to the pixel array.

4.7. Projection and distortion of points

At this stage, each point of the cloud is projected into the image, applying optical distortions, if any. The distortion model is applied to the undistorted normalized coordinates in the image plane $x = r_x^C/r_z^C$ and $y = r_y^C/r_z^C$.

An example of a commonly used model is the Brown distortion model [65,66] which returns the distorted normalized coordinates x' and y' based on three radial distortion terms k_1, k_2 and k_3 and two tangential distortion terms p_1 and p_2:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 2p_1 xy + p_2\left(r^2 + 2x^2\right) \\ p_1\left(r^2 + 2y^2\right) + 2p_2 xy \end{bmatrix} \tag{39}$$

where $r = \sqrt{x^2 + y^2}$. Then, the intrinsic camera matrix K is used to map the points in the pixel array:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f/\mu_u & s & c_u \\ 0 & f/\mu_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{40}$$

Applying distortions *before* the gridding allows an energy-conserving result in the final rendered image as the energy carried by each point is distributed to a location displaced consistently with the optical ray paths. Conversely, applying the distortion after the gridding would imply interpolation between pixels, leading to a lack of energy conservation in the image generation pipeline. This can be seen in Fig. 18, where four different distortions are applied to a texture-less sphere characterized by an area reflection model. The regions with a higher concentration of rays are brighter than the original values as the total energy focused on them is larger.

4.8. Direct gridding to pixel array

In this step, each value of the point cloud $(P/L)_j$ is assigned to the corresponding pixel in the image array $(P/L)_{u,v}$ at horizontal and vertical coordinates u and v, respectively. When shifting from the continuous point cloud domain to the discrete pixel domain, care must be taken to assign the correct intensity to the neighboring pixels. The main problem that is encountered during this operation is the phenomenon of *aliasing* and the appearance of *artifacts* in the final image. At the same time, the method should be energy-conserving to ensure radiometric consistency before and after the gridding.

The fragment shader in a typical rasterization pipeline interpolates the values at the vertices of the primitive surface to return the value at the center of the pixel. Conversely, in this work, the scattered data points refer to the midpoint of the quadrangle, and the values already embed the full radiometry contribution of that surface. In this case, *direct gridding* techniques can be used as they allow binning values from highly sampled scattered data points into a discrete grid. These techniques have been used for the generation of DEM from LIDAR data [67] and for image reconstruction in the remote sensing field [68]. Note that direct gridding does not interpolate data: the total sum of the values before and after gridding remains constant, and this ensures energy conservation.

The simplest direct gridding method is nearest-neighbor gridding, where the value of each point is assigned to the closest pixel. Although fast, this method gives rise to aliasing due to spherical geometry and the discrete nature of the problem. This is confirmed by the rendering in Fig. 20(a), which suffers from visible ringing artifacts. For this reason, the following alternative techniques were developed:

- **Weighted gridding**. A novel gridding method is proposed that splits the values of each point to the neighboring pixels according to the percentage of quadrangle area that, once projected into the image plane, falls into each pixel. The concept is depicted in Fig. 19(a). The weights must be computed such that their sum always equals 1 for energy conservation. By calling d_{u_k} and d_{v_k} the horizontal and vertical distances of the point j from the center of $k = 1, \dots, N_w$ neighboring pixels, two different weighting algorithms are proposed:

  1. **Area weighting**. It assumes the quadrangle projection as a 1-pixel size square centered on the point. The weights are computed only on 4 neighboring pixels and given by the fractions of the square area that falls on each pixel:

  $$\hat{w}_{jk} = \begin{cases} \left(1 - d_{u_k}\right)\left(1 - d_{v_k}\right) & d_{u_k} < 1 \quad \text{or} \quad d_{v_k} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{41}$$

  2. **Gaussian weighting**. It assumes a Gaussian distribution of the value centered on the point. Initial weights are calculated by evaluating the distribution function at the location of the neighboring pixels:

  $$w_{jk} = \frac{1}{2\pi\sigma^2} e^{-\frac{d_{u_k}^2 + d_{v_k}^2}{2\sigma^2}} \tag{42}$$

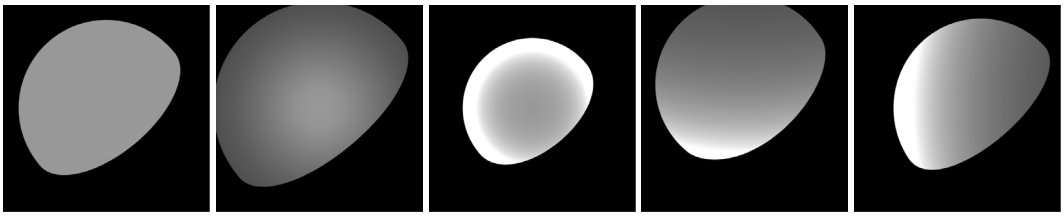
  and then normalized to the sum of all the weights to ensure energy conservation:

  $$\hat{w}_{jk} = \frac{w_{jk}}{\sum_{k=1}^{N_w} w_{jk}} \tag{43}$$

  The standard deviation is chosen by default so that the 3σ is at the edge of the window.

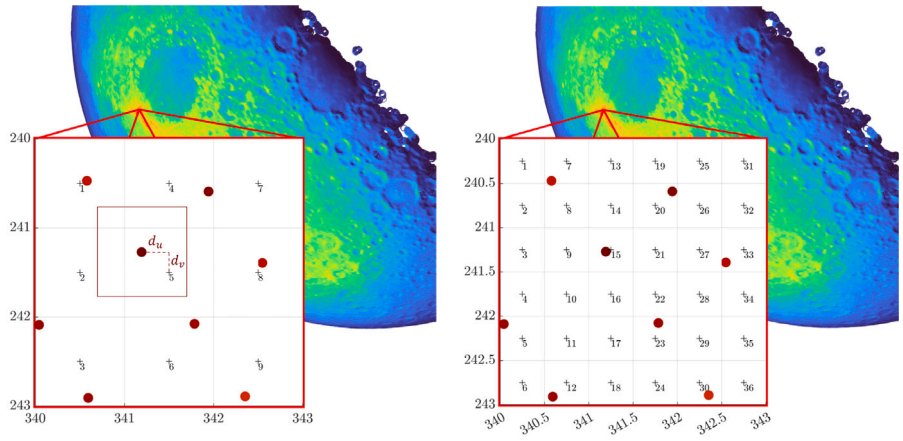
Once the weights to the neighboring pixels are computed for all the points, the value at the pixel (u, v) is obtained with a weighted conditional sum:

$$(P/L)_{u,v} = \sum_{\substack{j \\ \lfloor u_j \rceil = u \\ \lfloor v_j \rceil = v}}^{N_p} \left(\sum_{k=1}^{N_w} \hat{w}_{jk} (P/L)_j \right) \tag{44}$$



(a) Undistorted (b) Pincushion (c) Barrel (d) Vertical decenter (e) Horizontal decenter

Fig. 18. Distortion models applied to a sphere with constant albedo and an Area reflection model. Brighter or darker regions are not caused by reflection geometry, but by the projection of the rays falling more or less concentrated than in the case without distortions.



(a) Weighted gridding. With no weight (nearest-neighbor), the entire value would be assigned to the pixel labeled with 5. With area weighting, the value would be split to pixels 1, 2, 4, and 5. With Gaussian weighting, all the 9 pixels would receive a fraction of the value.

(b) Supersampling. The gridding is run on a virtually increased resolution grid. Then, the upsampled matrix is resized to the original resolution using a reconstruction filter.

Fig. 19. A Direct Gridding method is used to assign the values of scattered data points to a discrete grid. Weighted gridding enables splitting the value into the neighboring pixels proportionally to the area that the point occupies. Super-sampling can be used to virtually increase the resolution of the grid.

where $\lfloor u_j \rceil = u$ and $\lfloor v_j \rceil = v$ enforce the requirement for the points discrete coordinates u_j and v_j to fall inside the corresponding integer pixel (u, v).

- **Super-sampling**. Super-sampling is an anti-aliasing technique inherited from the computer graphics field [69]. The gridding algorithm runs on a pixel array up-sampled with a given granularity. Then, the image is resized to the correct resolution with a reconstruction filter such as Box, Bilinear, Lanczos, or Gaussian filters. Before resizing, the image can be convolved with an anti-aliasing Gaussian kernel. The concept is depicted in Fig. 19(b). A more accurate distribution of the values of each point is achieved as the granularity increases, since we get closer to the continuous domain. However, the higher the granularity, the larger the pixel array to process and the RAM required.

At parity of discretization points, artifacts can be effectively removed using one or more of these methods. Super-sampling has a milder effect (see Fig. 20(b)), while weighted gridding is more effective (see Fig. 20(c)), especially when using the Gaussian weighting method (see Fig. 20(d)), where no artifacts are visible. It is worth noting that both methods can be combined at the same time, depending on the image resolution and RAM constraints.

Before the application of detector noises, optical effects such as defocusing and aberrations caused by the finite Modulation Transfer Function (MTF) could be applied. These effects almost always make real images blurrier than synthetic equivalents. They are typically modeled via Point Spread Function (PSF) convolution [70] on the final image. Nevertheless, a Gaussian PSF may be mimicked in this step as a first approximation by tuning the size of the window and the standard deviation employed in the Gaussian-weighted Direct Gridding algorithm. More complex effects have not been implemented, but can be included either at this stage of the rendering or when post-processing the final image, without affecting the overall pipeline.

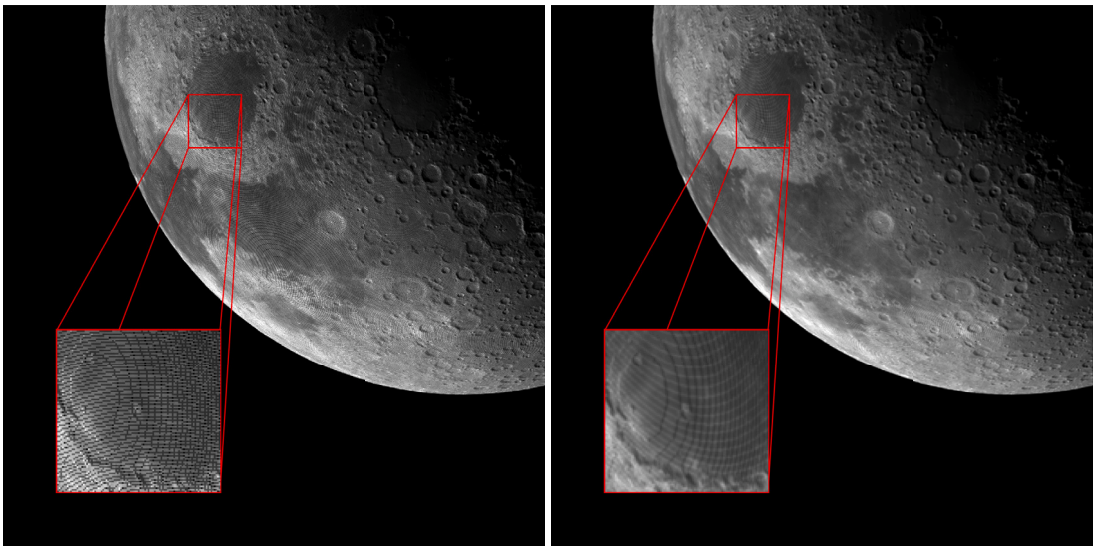
4.9. Noise and image generation

The last step in the rendering procedure is the generation of the image as an array of quantized DN values. To do so, the pixel-wise signal in terms of electrons per second at that waveband is first computed, in accordance with Eq. (10), using:

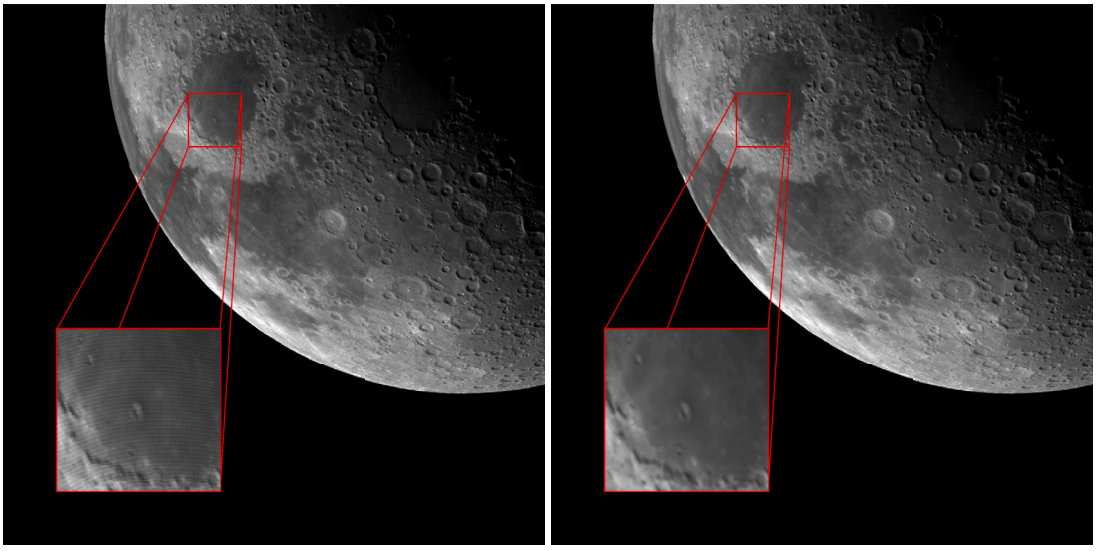
$$S_{u,v,\Delta\lambda} = (P/L)_{u,v} \int_{\Delta\lambda} \mathrm{QE}(\lambda)\, \mathrm{T}(\lambda)\, \frac{\lambda}{hc}\, L_{e\lambda}\, \mathrm{d}\lambda \tag{45}$$

For a multi-channel image, if $(P/L)_{u,v}$ is spectral-independent, the values in each channel can be obtained simply by computing the correct spectral integral along the corresponding waveband, speeding up the rendering process. Otherwise, $(P/L)_{u,v}$ has to be computed for every waveband separately.

Before applying time integration and quantization of the signal, the array can be processed with various sources of detector-dependant noises, which work on the raw electron content rather than on the quantized DN value. A non-exhaustive list of the most important ones are herein reported:



(a) Nearest-neighbor (b) Supersampled nearest-neighbor



(c) Area-weighted (d) Gaussian-weighted

Fig. 20. Rendering of the Moon using different direct gridding methods.

- **Photon Shot Noise**. It models a random variation in the number of photons detected during the exposure. Usually, it is modeled by a Poisson distribution with variance equal to the number of electrons collected in each pixel [71].
- **PRNU**. It defines a different photo-response among the pixels and is often modeled with a multiplication factor of the electron count that is normally distributed around one [71].
- **Dark Current**. It is a function of temperature and exposure time. It has a fixed-pattern component, which may be estimated at calibration with the generation of a *master frame*, and a time-dependent one, which is again characterized by a Poisson distribution on each pixel with variance equal to the dark current charge collected [71]. In the absence of calibration data, the fixed-pattern component can be modeled with a Gaussian distribution [72].
- **Readout Noise**. It arises from various sources in the electronics involved during the reading of the signal. It is often assumed to have a Gaussian distribution centered on zero to model the over and under-reading of the signal [71]. Minor effects such as flicker and row/column noises are not treated here due to their small influence and low understanding [71].
- **Motion Blur**. This effect is caused by the relative motion of the objects in the scene with respect to the camera during the integration time, which causes point light sources to create streaks in the images and blur extended objects. This effect can be modeled by stacking multiple renderings taken at shorter exposure times than the original one, one frame over the other.
- **Blooming**. This effect arises at saturation when the electron count in a pixel overcomes the Full Well Capacity (FWC) limit. If this happens, the electrons that are collected over this limit may spread to the neighboring pixels, creating a sort of halo around the saturated region and increasing the apparent size of the imaged object. This is a highly non-linear effect, and few sources have been found in the literature that attempted to characterize it [4,73,74]. Sources on existing models are even less, and restricted to unresolved objects [75].

There exist many other detector and electronic artifacts — smearing, rolling shutter, residual images, ghosts, electromagnetic interference,

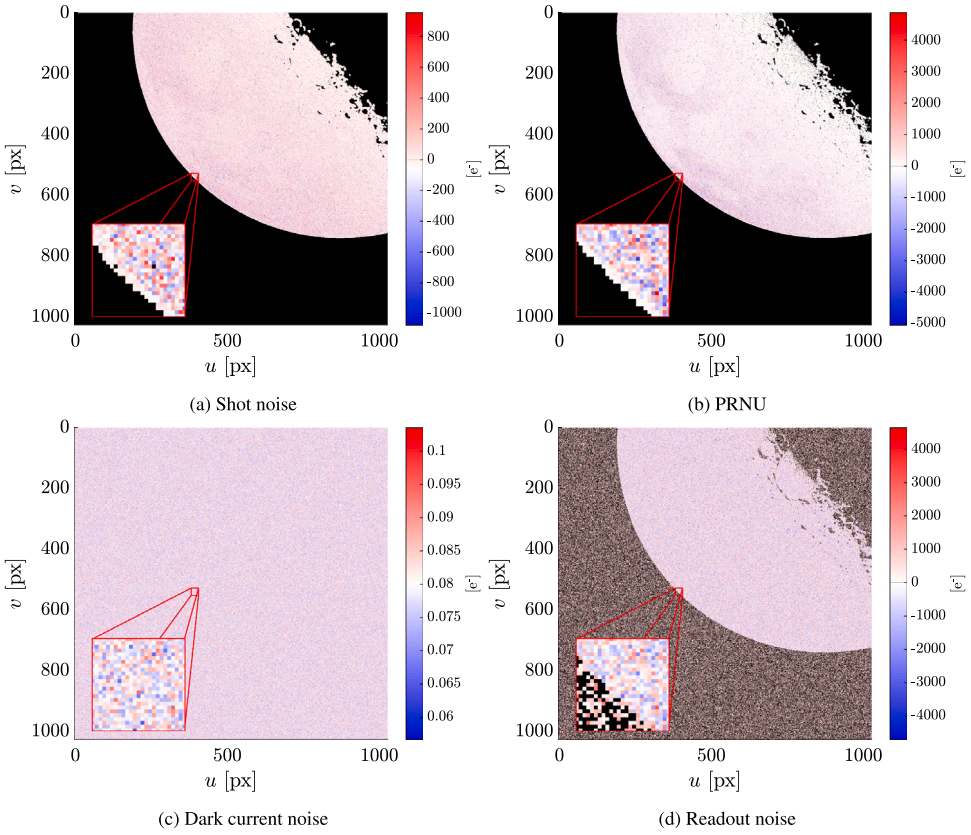


Fig. 21. Effect of different noises on the image as additive electron masks. Shot noise and PRNU affect only charge-collecting pixels, whereas dark current and readout noises act on the entire pixel array. Black pixels correspond to zero noise.

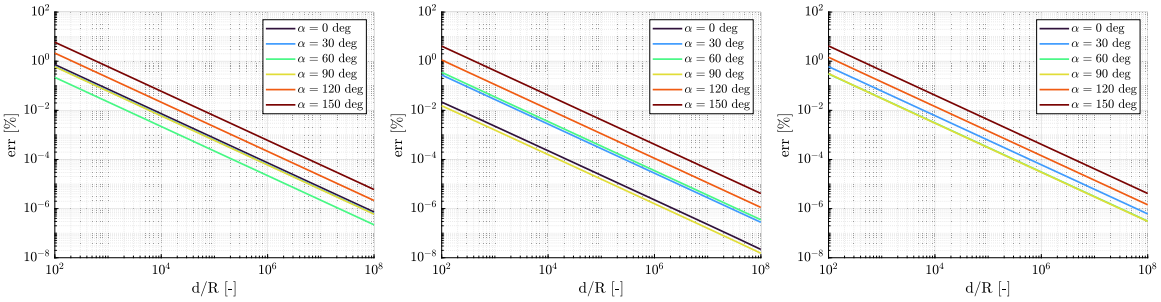


Fig. 22. Validation of the methodology against phase laws at different phase angles using three reflection models. The plots show the difference between the expected value from the phase law and the one measured from the image. The rendering would perfectly match the phase law only at infinite distance, where the phase law assumptions are verified.

among others. Being highly camera-dependent (varying by sensor type like CCD/CMOS, readout architecture, and shielding), they have not been thoroughly investigated and can be treated as tailored features to be added, if needed, to the consolidated pipeline.

The impact of several noise sources on the electron content of the image for an example scenario is shown in Fig. 21. The noise masks are additive, and they stack on the actual scene electron signal *before* proceeding with the analog-to-digital conversion.

After application of noises, the signal is integrated in time and capped between the zero-level and the FWC:

$$e^-_{u,v,\Delta\lambda} = \max\left(0,\ \min\left(\mathrm{FWC},\ \int_{\Delta t} \tilde{S}_{u,v,\Delta\lambda}\ \mathrm{d}t\right)\right) \tag{46}$$

where $\tilde{S}$ is the processed signal in electrons per second after application of the aforementioned effects.

The electron array is then converted to the DN values in the desired bit depth, using the Analog-to-Digital (AD) gain and offset. Finally, the closest unsigned-integer rounding is carried out to retrieve the quantized value, and the result is capped again to the converter bit depth:

$$\mathrm{DN}_{u,v,\Delta\lambda} = \max\left(0, \min\left(2^{N_{bits}} - 1,\ \left\lfloor \mathrm{DN}^{(0)} + G_{AD} \cdot e^-_{u,v,\Delta\lambda} \right\rceil\right)\right) \tag{47}$$

The image is now available as an array of quantized values in the desired bit depth and within the desired waveband. A multispectral image can be obtained by stacking the different images on each channel.

5. Validation

For a physically based rendering methodology, geometric and radiometric validation are of the utmost importance to generate reli-

able images for the training and testing of algorithms. The validation pipeline of this work follows a bottom-up approach. First, ground-truth analytical models are used to validate simplified scenarios, then the rendering validation complexity increases up to an end-to-end comparison against real images of the Moon acquired in space and from the ground.

5.1. Validation against phase laws

Analytical phase laws are available for several reflection models that embed the signal variation with respect to the phase angle. These phase laws are derived under two fundamental assumptions [52]:

1. **Far Distances**. The reflecting body is far from the light source and the observer. This means that the incoming and reflected rays that hit and bounce from the surface are parallel to the directions linking the center of the body with the light and camera, respectively. This assumption also implies that there are no unobservable regions due to tangency. A close-range observer placed at the equator would not be able to see the polar regions. Instead, when placed far away, those regions would be visible, and they would contribute to the energy content.
2. **Homogeneous Ideal Sphere**. The reflecting body is an ideal zero-displacement sphere with constant surface parameters, such as the albedo.

These assumptions – often made in planetary photometry – allows to write the phase law as a function of phase angle only. Analytical phase laws for three reflection models are reported hereunder:

$$\begin{aligned} \Psi_{Lambert} &= 1/\pi\,((\pi-\alpha)\cos(\alpha)+\sin(\alpha)) \\ \Psi_{Area} &= 1/2\,(1+\cos(\alpha)) \\ \Psi_{Lommel} &= 1+\sin(\alpha/2)\tan(\alpha/2)\log(\tan(\alpha/4)) \end{aligned} \tag{48}$$

A first validation could then be accomplished by comparing the total radiometric signal extracted from the rendered image with the one returned by the full sphere integrated phase law. This would check the correctness of the BRDF implementation and the validity of the discretized integration and gridding approach. To remain under the assumptions mentioned above, a texture-less sphere placed at 1 Astronomical Unit (AU) from the Sun is rendered at increasing distance-to-radius ratios from the camera. The *measured* phase law can be obtained by extracting Ψ from Eq. (11) and using the total radiometric signal (i.e., the sum of the values of the $(P/L)_{u,v}$ matrix):

$$\Psi_{meas} = \frac{\sum (P/L)_{u,v}}{A_{sun}\,A_{cam}\,\frac{\cos\epsilon}{d_{sun}^2 d_{cam}^2} R^2\,p_G} \tag{49}$$

If the measured phase law matches the expected value, it means the total radiometry signal is consistent and conserved throughout the rendering procedure.

The comparison is shown in Fig. 22 for the three phase laws at different phase angles, in terms of relative absolute error with respect to the expected analytical phase law value. At high d_{cam}/R ratios, the far distance assumption is increasingly justified and consequently the error drops. At closer range, phase laws are not generalizable for the reasons explained previously, and only the rendering is able to capture the correct amount of light collected by the camera.

From this analysis, we can conclude that the total radiometric content delivered by the rendering is consistent with the one expected from the reflection models used.

5.2. Validation against analytical model

An additional step in the validation pipeline is the comparison of the pixel-wise value expected from an analytical model based on the well-known *camera equation* [39]. According to this equation, the irradiance incident on the image E_i is proportional to the radiance reflected by the scene L_r:

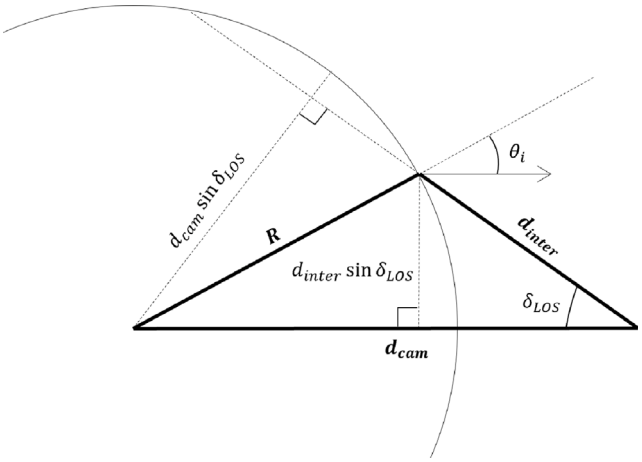


Fig. 23. Intersection of a Line-of-Sight (LOS) with a sphere. A triangle can be identified from the sphere radius R, the distance d_{cam} of the camera from the sphere center, and the distance d_{inter} of the camera to the LoS intersection point.

$$E_i = L_r\,A_{cam}\,\frac{\cos^4(\delta_{LOS})}{f^2} \tag{50}$$

In particular, the irradiance falls off with the fourth cosine of the angle δ_{LOS} between the optical axis and the imaged object, i.e., the angle of the LoS of the pixel.

Based on this equation, an analytical model can be derived that gives the irradiance collected by each pixel from an on-axis Lambertian sphere with constant albedo and at zero phase angle. The reflected radiance for this scene is a function of the incidence angle θ_i of the light on the sphere surface, which is in turn dependent on δ_{LOS}:

$$L_r(\delta_{LOS}) = \frac{p_N}{\pi}\,F\,\cos\left(\theta_i(\delta_{LOS})\right) \tag{51}$$

where the BRDF of a Lambert model $\frac{p_N}{\pi}$ has been plugged in. The incidence angle can be obtained with trigonometric considerations, as sketched in Fig. 23. The distance of the camera to the LoS intersection point d_{inter} is firstly obtained with:

$$d_{inter} = d_{cam}\,\cos\delta_{LOS} - \sqrt{R^2 - d_{cam}^2\,\sin^2\delta_{LOS}} \tag{52}$$

Then, the cosine of the incidence angle is computed with:

$$\cos\left(\theta_i(\delta_{LOS})\right) = \frac{d_{cam} - d_{inter}\,\cos(\delta_{LOS})}{R} \tag{53}$$

At this point, Eqs. (50), (51) and (53) are combined, and the irradiance is multiplied for the collecting pixel area A_{px} to obtain the power collected by each pixel as a function of the LoS angle only:

$$P(\delta_{LOS}) = A_{px}\,\frac{p_N}{\pi}\,F\,\cos\left(\theta_i(\delta_{LOS})\right)\,A_{cam}\,\frac{\cos^4(\delta_{LOS})}{f^2} \tag{54}$$

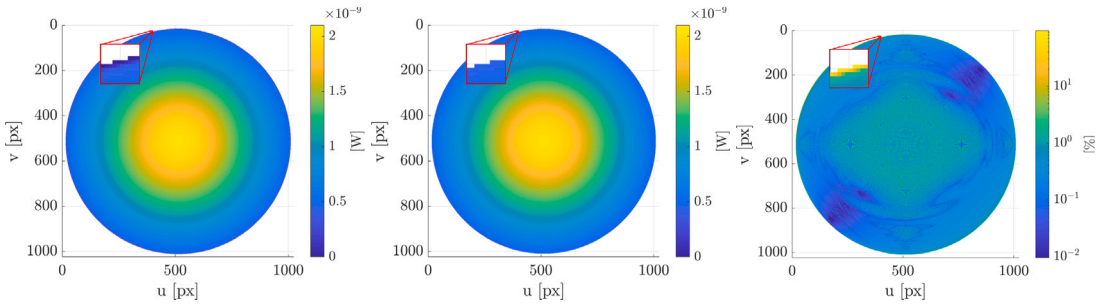
where F is the solar flux density at 1 AU for this particular scenario. The LoS angle is mapped from each pair of pixel coordinates u and v:

$$\delta_{LOS} = \arctan\sqrt{\left((u-c_u)\frac{\mu_u}{f}\right)^2 + \left((u-c_v)\frac{\mu_v}{f}\right)^2} \tag{55}$$

Combining Eqs. (54) and (55) an analytical image is obtained in terms of collected power for each pixel (u,v).

As per the image to validate, the *measured* power can be computed by plugging the matrix of $(P/L)_{u,v}$ coefficients in Eq. (9):

$$P_{meas}(u,v) = (P/L)_{u,v}\int_{\Delta\lambda} L_{e\lambda}\,\mathrm{d}\lambda \tag{56}$$



(a) Output from the rendering (b) Expected from analytical model (c) Relative Error

Fig. 24. Validation of the methodology against an analytical model of the collected power per pixel. To capture the 4th power variability, a wide-angle camera with $FOV = 85\,\text{deg}$ was used. The rendering can capture details at a sub-pixel scale.

The images of collected power and their relative error are shown in Fig. 24. High values of relative errors are only found on the limbs, where the rendering can capture the steep decrease in brightness at the sub-pixel level, not visible in the analytical model being accurate only up to the pixel scale. The image median relative error across all the pixels is under 0.3%. The difference in the total power collected (sum of the values of all the pixels) is less than 0.002%. This implies that local differences are only due to minor artifacts, with the total energy collected practically equivalent.

From this analysis, it can be stated that the rendering procedure in this idealized scenario can deliver the correct radiometric content and light distribution, both locally (pixel-wise) and globally.

5.3. Validation against real images

A last necessary step in the validation campaign is a direct end-to-end comparison of synthetic images against real images, either captured by a previously flown mission or taken directly from the ground. The Moon is chosen as a validation target due to its well-known geometric and radiometric characterization, implying knowledge of its surface parameters.

For what concerns space-borne images, the SMART-1 mission [76, 77] is selected since it has a large image database that includes hundreds of pictures acquired in different poses and labeled with exposure time and epoch metadata. Moreover, SPICE kernels are available to link the epoch of each image with the camera pose and the Moon orientation. Each image is extracted from the database following the procedure depicted in Fig. 25. From the image metadata, the camera temperature T, the exposure time Δt, and the current epoch t are extracted. The raw image is corrected by subtracting the bias and dark current, computed from the calibration master frames B and DC:

$$\tilde{\text{DN}}_{10} = \text{DN}_{10} - \left(\text{DN}_{10}^{(0)} + (B + DC \cdot \Delta t) \cdot f(T)\right) \tag{57}$$

where $f(T)$ is function of the temperature of the detector and $\text{DN}_{10}^{(0)} = 8$ is the fixed gain offset in 10 bit depth as reported in Grieger [78].

From the epoch, the position and orientation of the AMIE camera and the Moon are extracted using SPICE routines and used as input configuration files to the render engine. The camera parameters are read from SPICE kernels – if available – or extracted from mission documents otherwise. For example, Fig. 26 reports the transmittance T trend across the AMIE bandwidth retrieved from a hand-written calibration datasheet[10] and its piecewise interpolation fitting used in the rendering. A Hapke reflection model is considered for the rendering, with the parameters derived in Sato et al. [51].

The comparison for 5 example images [11][12][13][14][15] at different distances, phase angles, and exposure times is shown in Fig. 27. To be concise, each image has been labeled with a number. The parameters and label used for each image are reported in Table 3.

The AMIE images have been shifted due to uncertainties in the pose provided by the SPICE kernels, which inherently introduce errors in the comparison [79]. This problem is highlighted in Fig. 28 using as an example image AMIE2. The expected Moon shape coming from the SPICE kernels' state and orientation is plotted on top of the *real* image in Fig. 28(a), showing an error shift of some pixels that will be inherently carried after in the rendering. This shift is consistent with the one noticed in previous studies of the AMIE images [80]. To counteract this issue, the images are aligned by finding the optimal shift with Normalized Cross-Correlation (NCC). An *Alignment Picture* can be used to assess the quality of the alignment. The red channel of this picture contains the mask of raw image values that are above the noise threshold, while the blue channel contains the mask of rendered image values that are nonzero (i.e., light-collecting). If a pixel is colored magenta, it means the same pixel is active in both images. The comparison of the alignment masks without and with NCC alignment is shown in Fig. 28(c) and in Fig. 28(d), confirming the effectiveness of the technique.

Following this pre-processing, a quantitative assessment can now be performed. To check the similarity of the images, the Structural Similarity Index Measure (SSIM) [81] is evaluated in their *foreground*. In fact, we are not interested in the similarity of the background noise pattern, but rather in the consistency of the object texture shading and light interaction. Comparing the background would also cause a skewed positive effect on the structure figure of merit, being the background characterized by an almost-homogeneous flat gray shading variation for every image. The background is detected by selecting all pixels with an intensity lower than a threshold, which is different for each image. This threshold is computed as the median plus one standard deviation of the intensity values of the inactive pixels. Inactive pixels refer to those pixels in the real image that are located at the same positions as non-charge-collecting pixels in the rendered image.

An example of similarity assessment considering AMIE2 is shown in Fig. 29. The aggregated values for each image, obtained as the mean figures with one standard deviation uncertainty on all the active pixels, is also reported in Table 4. Luminance and contrast match pretty well (>95%) for all images, indicating consistency in the mean brightness and in the pixel intensity ranges between the real and synthetic images. On the other hand, the structure metric compares the spatial patterns, and therefore, the similarity is slightly lower, especially for image AMIE4. This difference can be traced back to the kernel state

[10] SPECTRAL_RESPONSE.PDF, last accessed in Dec. 2025.
[11] AMI_EE3_040819_00169_00010.IMG, last accessed in Dec. 2025.
[12] AMI_EE3_040819_00208_00030.IMG, last accessed in Dec. 2025.
[13] AMI_EE3_041028_00269_00005.IMG, last accessed in Dec. 2025.
[14] AMI_EE3_041111_00008_00040.IMG, last accessed in Dec. 2025.
[15] AMI_EE3_041111_00070_00018.IMG, last accessed in Dec. 2025.

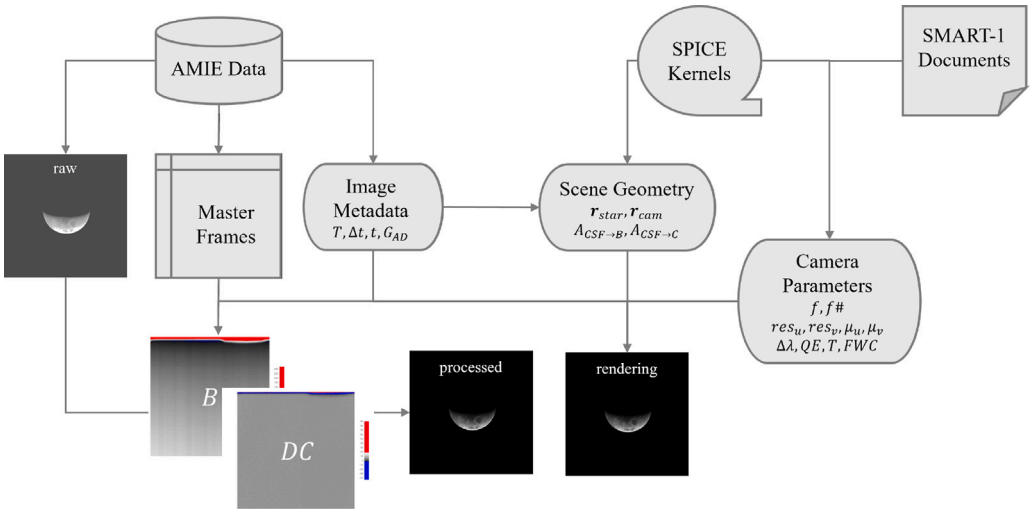


Fig. 25. Overview of the pipeline for extraction and processing of the AMIE images.

Table 3
Scene geometry and camera parameters for the real images validation campaign.

	Space-borne					Ground	
	AMIE1	AMIE2	AMIE3	AMIE4	AMIE5	BFS1	BFS2
Phase angle [deg]	106	105	7.5	144	142	50	51
Distance-to-radius [–]	114	114	387	49	47	218	218
Exposure time [ms]	10	30	5	40	18	0.114	0.08
Detector temperature [K]	286	286	297	289	290	N/A	N/A
Focal length [mm]	154.7	154.7	154.7	154.7	154.7	50	25

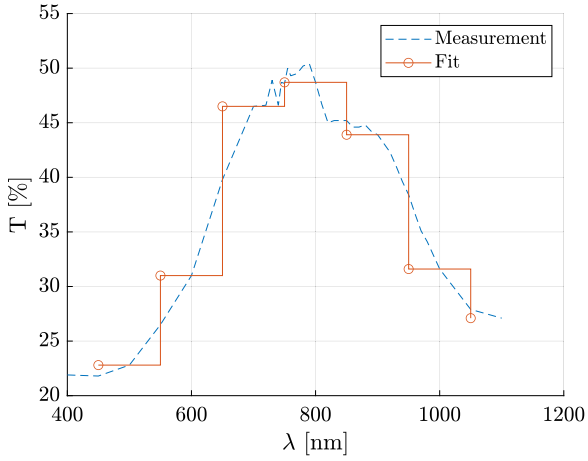


Fig. 26. Transmittance of AMIE camera.

inaccuracies, which could not be completely eliminated by the NCC alignment.

Additionally, an end-to-end validation using real images acquired on-ground is carried out. The sensor used is a Blackfly FLIR BFS-U3-31S4[16] camera body coupled with two different optical systems: a 50 mm- and a 25 mm-focal-length objective. The images were taken in northern Italy during a clear-sky night to reduce at minimum the degradation on the image quality due to the atmosphere. From the known camera location longitude and latitude on Earth and acquisition epoch, the ephemeris of the Sun and the Moon were extracted and plugged in as inputs to the rendering. Two example images are shown compared to their rendered counterparts in Fig. 30. Again, very good matching (>95%) is obtained in luminance and contrast, and good matching (about 85%) in structure. The higher difference in the structure may be traced back to light scattering and the blurring effect caused by the atmosphere. All in all, the SSIM metrics for the ground images validation campaign (see Table 4) are consistent with the ones found within the space-borne images validation.

Table 4
Aggregated SSIM metrics (mean and standard deviation) for the real images validation campaign.

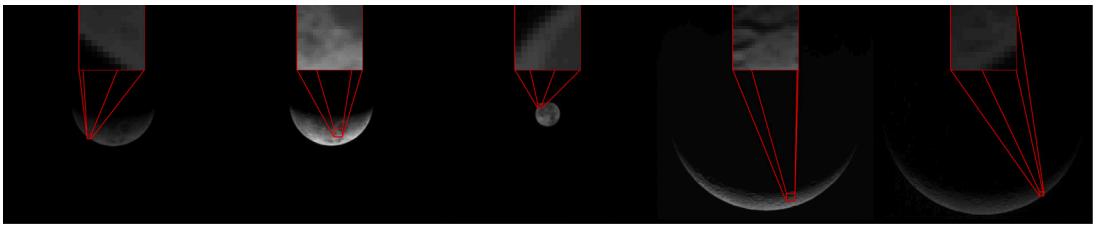
	Space-borne				
	AMIE1	AMIE2	AMIE3	AMIE4	AMIE5
Luminance	0.981 ± 0.04	0.980 ± 0.05	0.982 ± 0.06	0.967 ± 0.05	0.968 ± 0.05
Contrast	0.996 ± 0.01	0.982 ± 0.04	0.983 ± 0.04	0.958 ± 0.06	0.986 ± 0.02
Structure	0.988 ± 0.02	0.940 ± 0.09	0.985 ± 0.03	0.853 ± 0.17	0.955 ± 0.06
	Ground				
	BFS1	BFS2			
Luminance	0.983 ± 0.07	0.963 ± 0.10			
Contrast	0.962 ± 0.06	0.951 ± 0.07			
Structure	0.880 ± 0.11	0.881 ± 0.12			

Following this extended and incremental validation campaign, the rendering methodology is demonstrated to be validated against the available analytical models and real images from both a geometric and a radiometric point of view.

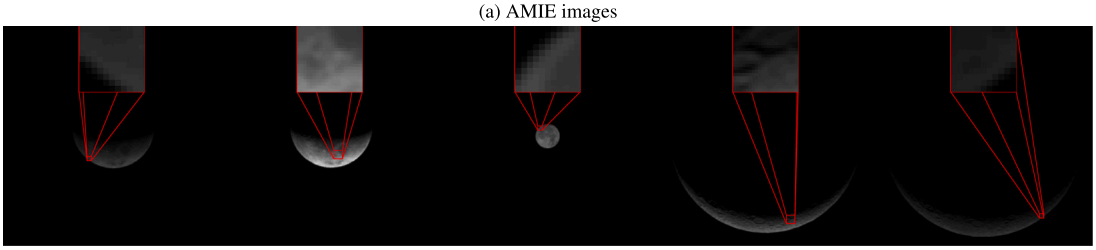
6. Results

This section will provide a brief overview of the capabilities of the render engine, followed by a benchmark against open-source and commercial tools. Finally, some examples of applications are given.

[16] BFS-U3-31S4 Spec, last accessed in Dec. 2025.

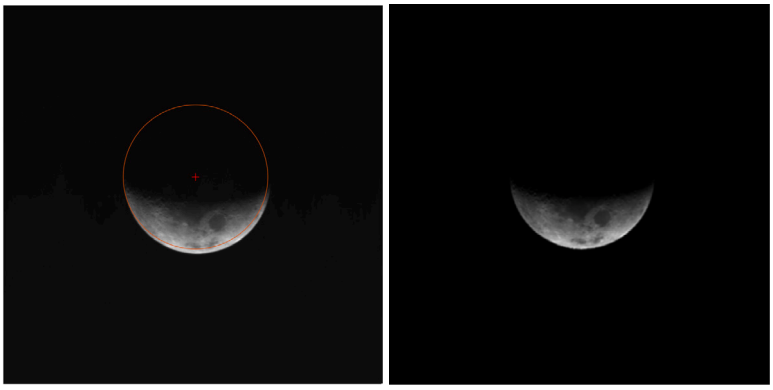


(a) AMIE images



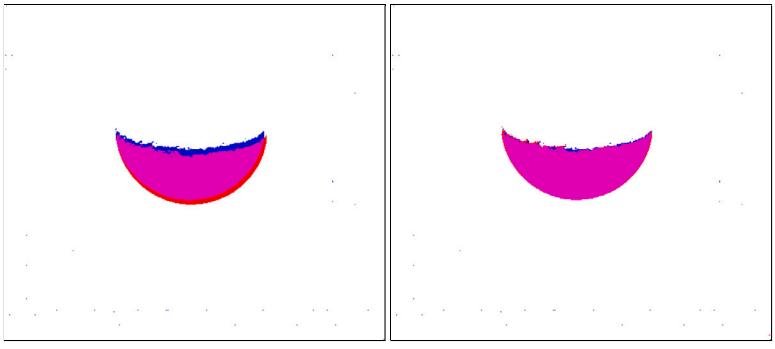
(b) Rendering

Fig. 27. Validation of the methodology against space-borne images. The AMIE images have been corrected for bias and dark current, and they have been slightly shifted to account for SPICE uncertainties. Zoomed insets have been overlaid to facilitate the reader in the qualitative assessment of rendering quality across features at different scales.



(a) Expected moon shape from SPICE kernels superimposed on real image.

(b) Rendering



(c) Alignment picture before NCC shift

(d) Alignment picture after NCC shift

Fig. 28. Alignment of space-borne images through normalized cross-correlation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6.1. Rendering capabilities

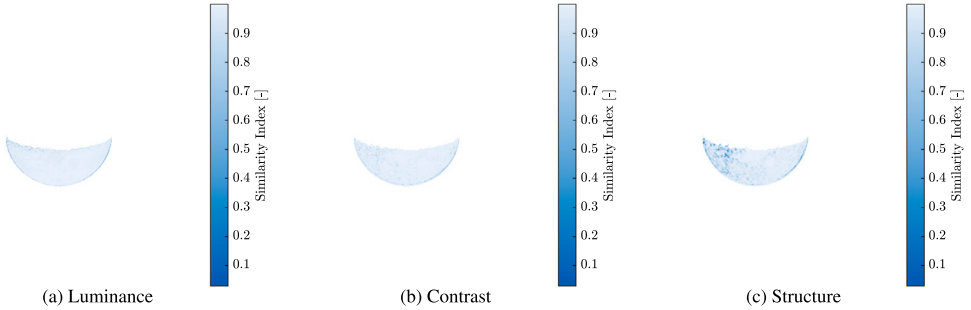
6.1.1. Close-range

Rendering examples have previously been shown for mid-range distances in Fig. 1 using as targets different planets and moons of the solar system. Being the Moon a target of interest for near-future human exploration missions, Fig. 31 depicts some renderings at a closer range for this particular scenario.

6.1.2. Chromatic aberrations

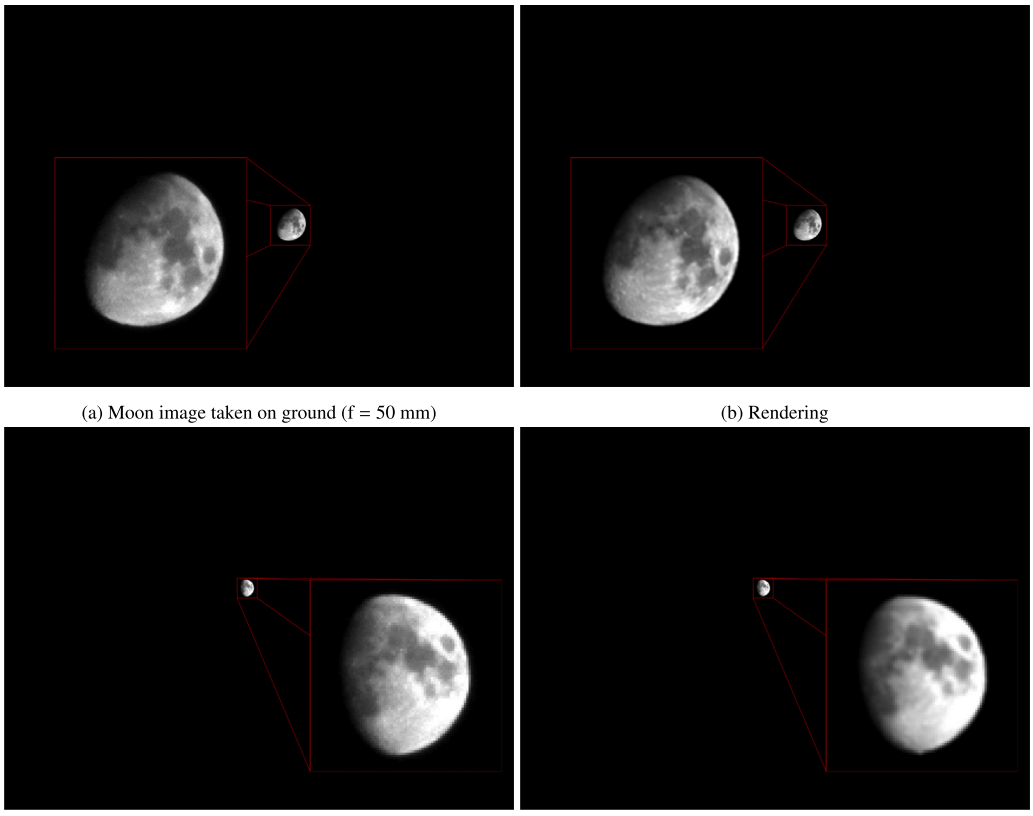
Building on the flexibility of the techniques and algorithms used within the methodology, chromatic aberrations can be included, provided the knowledge of relative differences of refraction and focusing at each wavelength.

On one hand, lateral chromatic aberrations can be modeled by modifying the distortion parameters used for projection (see Section 4.7) depending on the waveband. On the other hand, longitudinal chromatic



(a) Luminance (b) Contrast (c) Structure

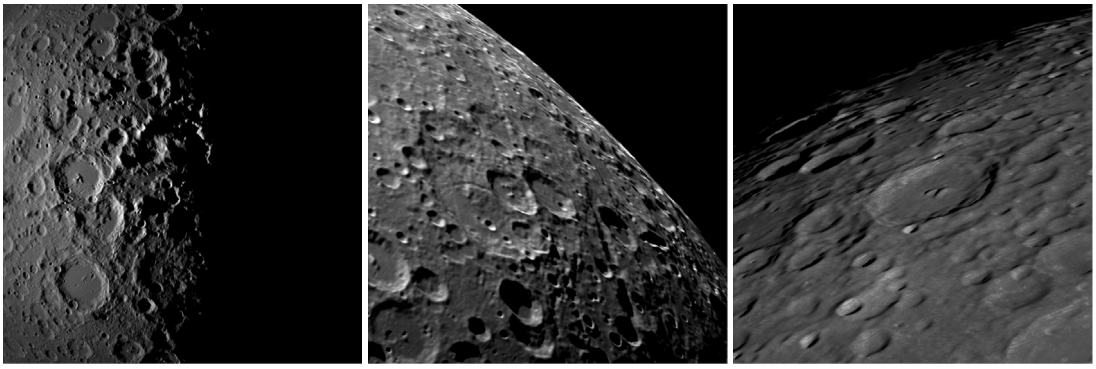
Fig. 29. Similarity assessment of image AMIE2 using SSIM. Luminance compares the overall brightness, Contrast measures the deviation from the mean intensity, and structure correlates the local patterns.



(a) Moon image taken on ground (f = 50 mm) (b) Rendering

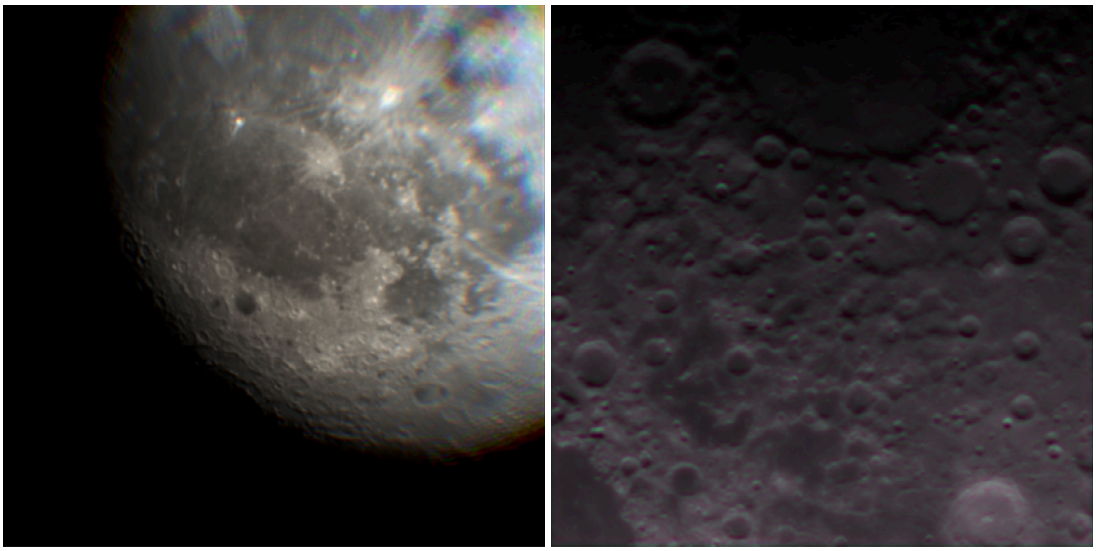
(c) Moon image taken on ground (f = 25 mm) (d) Rendering

Fig. 30. Validation of the methodology against ground images of the Moon taken during a clear-sky night in northern Italy.



(a) South pole (b) Off-nadir limb (c) Off-nadir terminator

Fig. 31. Moon at close-range distances. The high fidelity of shadows is noticed especially in the proximity of crater limbs.



(a) Lateral (b) Longitudinal

Fig. 32. Rendering of chromatic aberrations. The parameters have been exaggerated to make the effects noticeable to the reader. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

aberrations can be modeled as spectral-dependent focus shifts, by tuning the standard deviation and window size of the Gaussian-weighted direct gridding algorithm (see Section 4.8).

Examples of lateral and longitudinal chromatic aberrations are shown in Figs. 32(a) and 32(b). In the first image, the colored fringes at the corner radially increase in shift due to the relative change in distortion between the different ray colors, as expected [82]. In the second image, the focusing is different for each waveband, and as a result, red/violet regions are alternated with greenish ones due to the combination of the colors [82].

6.1.3. Small bodies

An interesting aspect to evaluate is the break point of the procedure when going away from the sphericity assumption. In fact, dwarf planets and asteroids like Ceres, Vesta, or Phobos are characterized by large displacement differences across their surface. In these cases, the DEM is defined with respect to a reference ellipsoidal shape or explicitly given in absolute meters from the center. Some rendering examples are shown in Fig. 33 compared against real images[17][18][19] captured by missions that flew to these bodies (i.e., Dawn and Viking). Note that in this case, no quantitative comparison was made, as established global albedo maps, reflection models, and parameters are not yet available for these bodies. This simple visual comparison only aims to show how far the model can be stretched from its assumptions without compromising the overall photorealism.

6.2. Benchmark

The characterization of the rendering performance is an important step that drives the choice of a tool over another. The higher the image generation frame rate, the smaller the time required to generate large datasets for algorithm training and validation. Moreover, in the case of closed-loop testing applications, a high frame rate allows a smooth and fast testing of algorithms in an integrated Hardware In the Loop (HIL) framework. At the same time, it is almost impossible to express with a single number the performance of a tool as this last largely differs depending on the scene, geometry, as well as the number of points or mesh size to process.

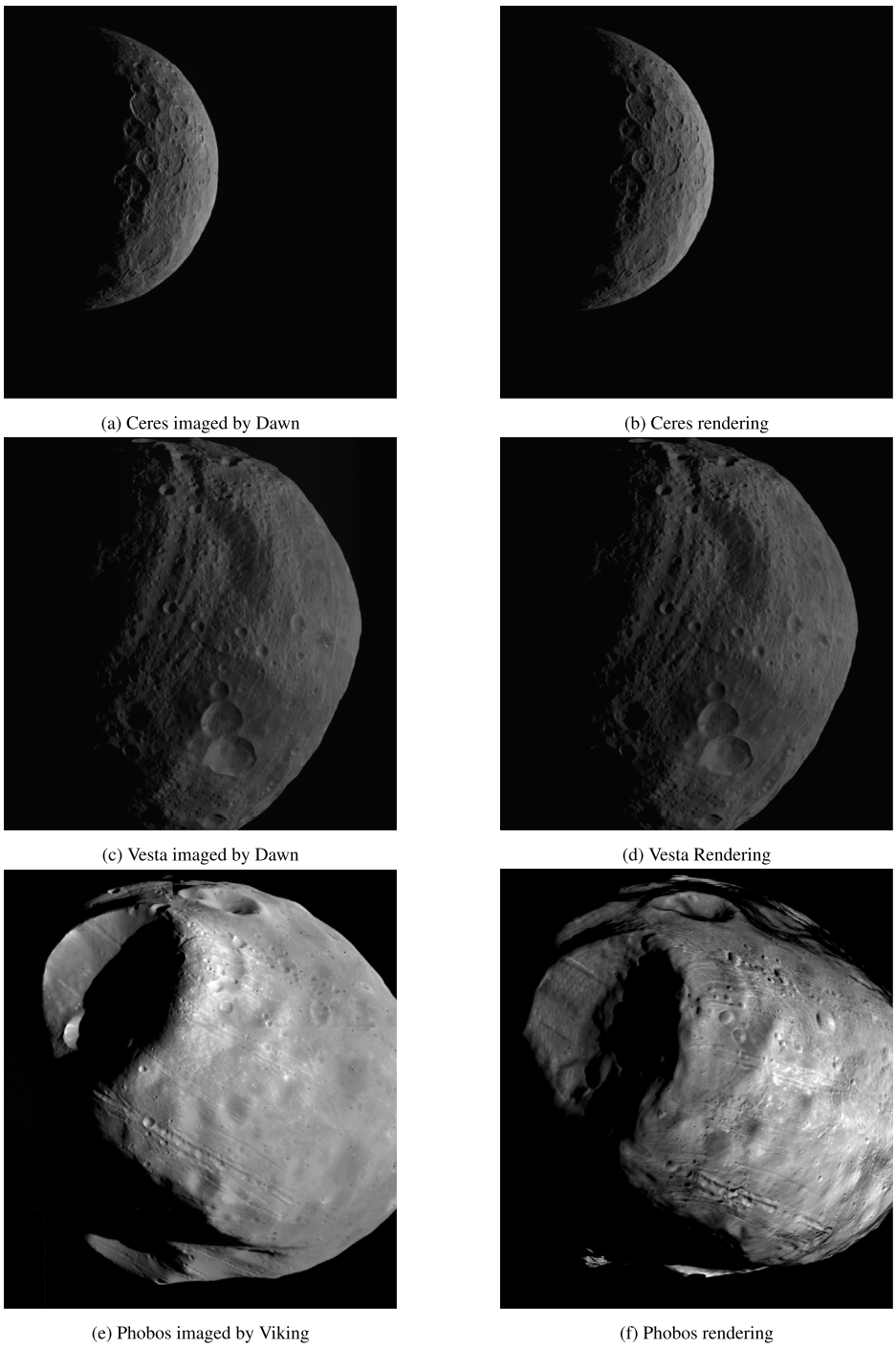
For this reason, it is common to compare the performance on a *benchmark* scenario. Within this article, a Moon orbit scenario is employed with large variations in distance, pointing, and phase angle. The attitude is set so as to have the Moon limb always in the FOV to allow optical navigation. The initial conditions corresponding to a Near-Rectilinear Halo Orbit (NRHO) are propagated for two orbits. This kind of orbit is the one expected for future Moon exploration missions [83].

The performance is benchmarked in terms of frame rate (FPS) against the ones provided by current state-of-the-art tools on the same trajectory. For the scenario of imaging of planets and moons, Blender is the natural competition choice, being widely adopted within the open-source space imaging community. To this aim, the Python interfaces available in CORTO [36] are used as they are convenient for generating the images programmatically, even if not optimized yet in terms of caching and memory management. For this comparison, a Lambert model is used as an Hapke model is not available in Blender naturally. The same global albedo and displacement maps are provided to both tools (16 pixel-per-degree resolution). Note that the normal map is not used in Blender, as shadows are inherently generated by how the ray-tracing technique works. The same camera parameters of the ground validation campaign are used, only changing the resolution to 1024×1024 px for a fair comparison against the frame rates declared by the commercial tools.

The rendering performance is reported in Fig. 34 compared to that of Blender Cycles using 128 rays per pixel on a common laptop CPU equipped with a 6-core AMD Ryzen 5 5600H (3.3 GHz). Frame rates spanning 1 to 10 Hz are achieved between mid and far range. The performance exceed Blender for the entire trajectory and are in line with the ones declared by commercial tools, such as SurRender (about 0.2 Hz on CPU for a Moon flyby scenario, an image size of 1024 pixels and 128 rays per pixels [8]) and PANGU (2–5 Hz for Mars in full FOV [10]).

When the Moon enters the full FOV or the pointing is strongly off-nadir, performance drops under 1 Hz due to the large number of discretization points to process. This is also highlighted in Fig. 35, which reports the fractions of time taken by each step of the

[17] FC21A0034721_15115071008F1I.FIT, last accessed in Dec. 2025.
[18] FC21A0003042_11204181903F1I.FIT, last accessed in Dec. 2025.
[19] Phobos-viking1.jpg, last accessed in Dec. 2025.



(a) Ceres imaged by Dawn

(b) Ceres rendering

(c) Vesta imaged by Dawn

(d) Vesta Rendering

(e) Phobos imaged by Viking

(f) Phobos rendering

Fig. 33. Real images of dwarf planets and asteroids compared to rendering. For Ceres and Vesta, the rendering reflects the real pose of the Dawn camera in that frame. The picture of Phobos is a composite montage of three separate images, hence the camera used for the rendering is different, and a manual alignment had to be made.

procedure as a function of the fraction of FOV covered by the Moon in the frame. We notice that this metric is proportional to the number of discretization points to process, which spans 35 thousand to 6.3 million for the frames where the Moon covers the entire frame or the orientation is largely off-nadir.

Despite the clear over-performance of the proposed method with respect to the ray-tracing-based technique, no drawbacks in accuracy and fidelity are found, as shown by the direct comparison of one of the closest-range frames in Fig. 36. Note that for this comparison, Blender had to be globally "radiometrically calibrated" using the procedure described in [17,84], being its DN values unlinked with physical reality. Nevertheless, for what concerns local features, shading and geometry are reproduced with very good consistency by the methodology with no visible artifacts.

For some frames, our method is even better than Cycles in rejecting artifacts and rendering morphology features. We notice this in the

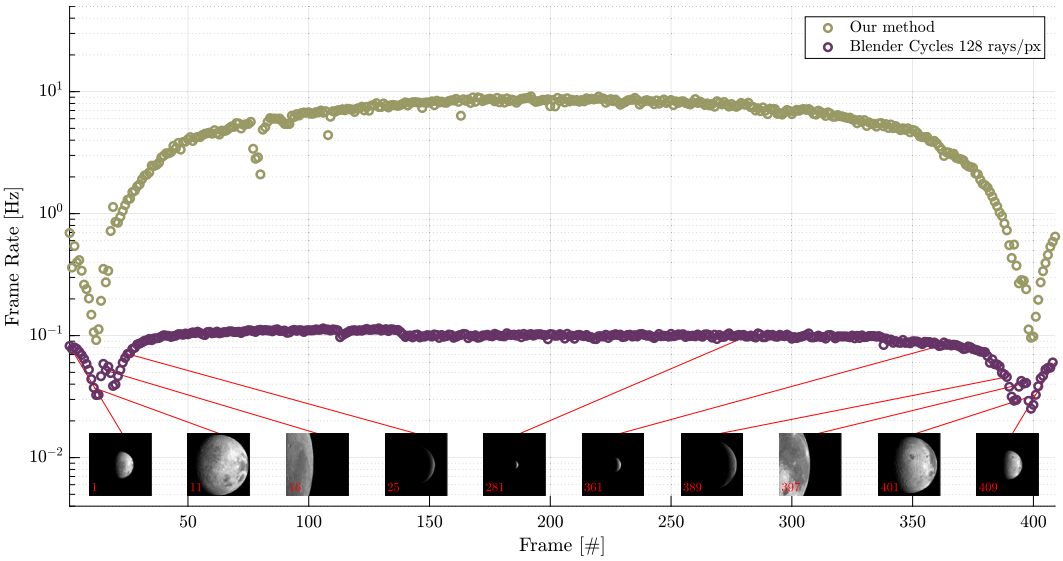


Fig. 34. Benchmark of the frame rate against Blender Cycles for a NRHO orbit around the Moon. The phase angle changes along the trajectory with excursions spanning from 40 deg to 150 deg, while the altitude covers a range from 1500 km to 70 000 km. Performance drops for both methods during close-range off-nadir phases.

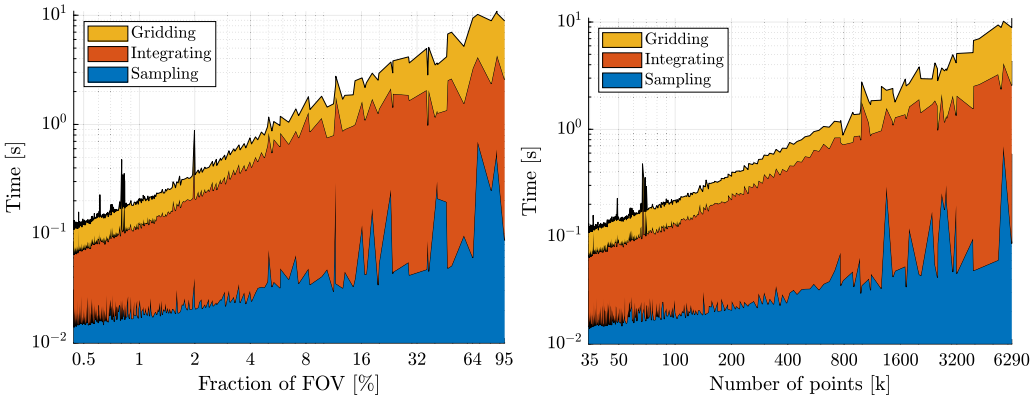
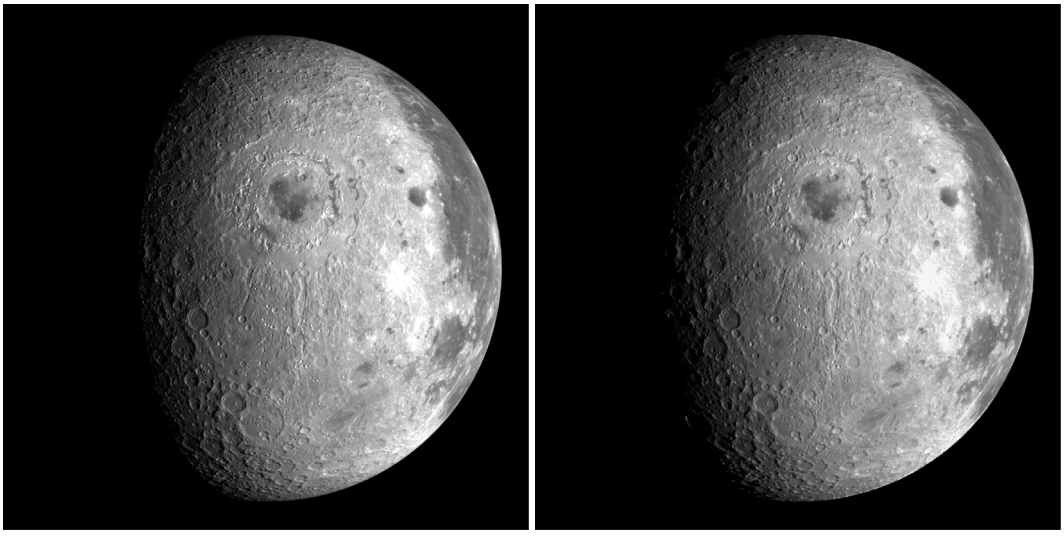
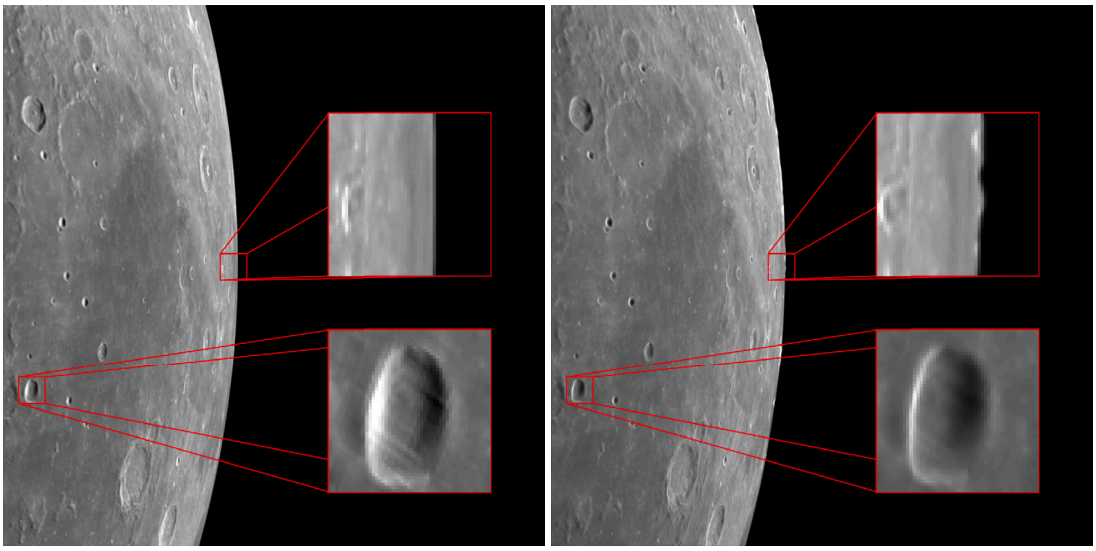


Fig. 35. Rendering time split between sampling, gridding, and integration steps as a function of FOV coverage and number of discretization points.



(a) Blender Cycles (after calibration) (b) Our method

Fig. 36. Comparison of Moon full disk rendered using Blender Cycles vs. our method.



(a) Blender Cycles (after calibration) (b) Our method

Fig. 37. Comparison of small-scale features on the Moon rendered using Blender Cycles versus our method.

zoomed boxes of Fig. 37, where minor artifacts and a straight horizon appear in the Cycles rendering. Note that if not correctly rendered, these features may lead to an inaccurate assessment of the performance of computer vision techniques such as limb-fitting navigation algorithms.

This benchmarking campaign reveals the tool is a competitive choice against state-of-the-art available techniques, especially when rendering objects at far and mid ranges. We highlight that the tool is coded in MATLAB as of now, and it is therefore not code-efficient like the competitors, so performance may be even improved.

6.3. Applications

The most straightforward use of this tool would certainly fall into the generation of images for the testing of vision-based navigation algorithms or for the training of data-driven techniques. However, the authors would like to highlight in this section some of the applications for which only a tailored render engine that outputs physical and radiometry quantities can be used. Note that this is a desirable property that often lacks in open-source render engines, as they are more addressed at the artistic community rather than the scientific one. Conversely, when this information is available, the rendering methodology can be used for two example applications: in parallel with an *HIL optical stimulator* to test algorithms with radiometric consistency, or to carry out camera design studies, for instance to tune the *optimal exposure time* for taking well-exposed pictures during the approach to a target known body.

6.3.1. HIL optical stimulator

An optical stimulator is an HIL test bench based on three main elements: a screen emitter, a camera, and a set of lenses/collimators placed between them. The camera, preferably identical to the one that will fly in the real domain [19], is stimulated by the screen. The lenses and collimators are used to focus and adjust the image ratio to the one expected [24,25].

When using an optical facility, it is of paramount importance to maintain geometric and radiometric consistency. In both cases, a calibration is required to extract the facility-induced distortions for the former and to compute the mapping between the DN that any pixel of the screen should emit to deliver the correct radiant flux density to the camera for the latter. The reader is redirected to Panicucci et al. [85] and Ornati et al. [86] for more details on this procedure.

After the calibration is accomplished, the facility can be used, but if and only if the image of *effective radiant flux density* at screen level is available. The effective radiant flux density as defined in Ornati et al. [87] is given by:

$$F_{\text{eff}} = \int_{\lambda} \frac{\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda}{\max_{\lambda}\left(\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda\right)} F_{\lambda}\,\mathrm{d}\lambda \tag{58}$$

Thanks to the radiometric calibration, this quantity is converted to screen DN, and the camera can be stimulated with the true radiometric content.

The effective radiant flux density can be written as a function of the (P/L) coefficient and of the signal S as:

$$\begin{aligned} F_{\text{eff}} &= \int_{\lambda} \frac{\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda}{\max_{\lambda}\left(\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda\right)} \frac{P_{\lambda}}{A_{cam}}\,\mathrm{d}\lambda \\ &= \frac{1}{A_{cam}} \frac{hc}{\max_{\lambda}\left(\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda\right)} \int_{\lambda} \frac{\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda}{hc} P_{\lambda}\,\mathrm{d}\lambda \\ &= \frac{hc}{A_{cam}\max_{\lambda}\left(\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda\right)} \underbrace{(P/L) \int_{\Delta\lambda} \text{QE}(\lambda)\,\text{T}(\lambda)\,\frac{\lambda}{hc} L_{e\lambda}\,\mathrm{d}\lambda}_{S} \end{aligned} \tag{59}$$

where Eqs. (8) and (10) have been used.

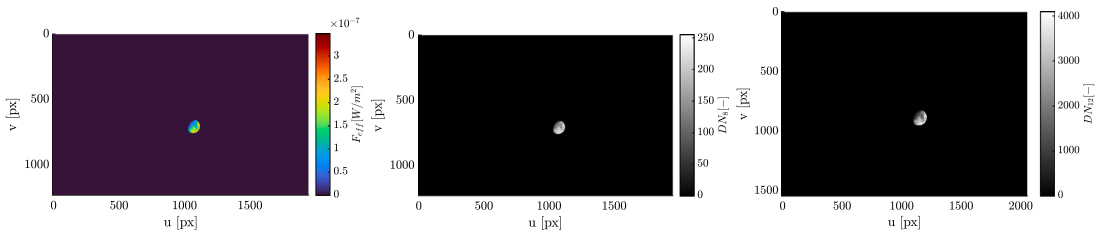
Ultimately, this means that to map the signal S with the matrix of F_{eff} the following scalar coefficient can be used:

$$k_{S\to F_{\text{eff}}} = \frac{hc}{A_{cam}\max_{\lambda}\left(\text{QE}(\lambda)\,\text{T}(\lambda)\,\lambda\right)} \tag{60}$$

Using the same setup of the ground image validation campaign (see Fig. 30), an end-to-end comparison is carried out between the rendered image, the real image, and the image as captured by the same hardware in the facility.

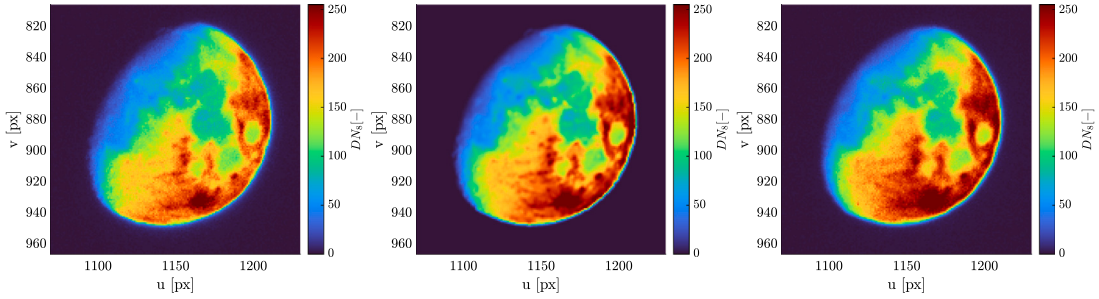
The RETINA optical facility available at DART Lab is used for this purpose [26]. The overall procedure is shown in Fig. 38. The following are the necessary steps to generate the image in the facility:

1. A matrix of F_{eff} is generated from the image rendered using the HIL facility parameters and the mapping coefficient of Eq. (60). Note that in general the HIL image has a different resolution, focal length, and pixel pitch with respect to the image rendered for the end-to-end comparison.
2. The F_{eff} matrix is converted to a screen image in DN through the radiometric calibration curve (see Ornati et al. [86] for more details).



(a) Matrix of F_{eff} (b) Image displayed on the screen (c) Image captured by the camera

Fig. 38. To use an optical stimulator with radiometric consistency, a matrix of effective radiant flux density must be generated and converted to screen DN so that the sensor can acquire an image with the correct radiometric content for each pixel.



(a) Real (b) Rendering (c) Optical Stimulator

Fig. 39. Zoomed comparison in false color scale of BFS1 image (see Section 5.3) generated in different domains. The optical facility returns the correct radiometric content thanks to the physically-based nature of the rendering methodology. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. The screen image is displayed on the screen of the facility, and the camera acquires an image. This image has the desired radiometric content and the correct resolution, as the hardware used is the same of the one used for the night-sky image.

Considering the BFS1 image (50 mm) of the ground validation dataset, a HIL image is generated in RETINA and compared to the real and rendering counterparts in Fig. 39, showing consistency and matching. A small focus difference is noticeable close to the limb between the image acquired in the facility and the rendering, traceable to unmodeled optical aberrations in the latter. This claim is confirmed by the presence of blurring in the real image, where these effects are inherently present being the sensor used to capture the real image the same one deployed in the facility.

It is important to stress that the generation of physical quantities associated with the scene (i.e., the signal S in electrons per second) is necessary to use the optical facility with radiometric consistency, hence the need to have a rendering tool such as the one presented in this work.

6.3.2. Optimal exposure time

Picture planning during the approach to a target body is a result of cooperation between the engineering and science teams. Navigation performance informs the design of the science plan, but the science objectives may be met only if enough usable pictures are gathered [88]. Exposure times are typically preset in tabulated brackets according to the predicted camera relative pose to the target and the different engineering needs, such as limb extraction, feature detection, or background star field processing.

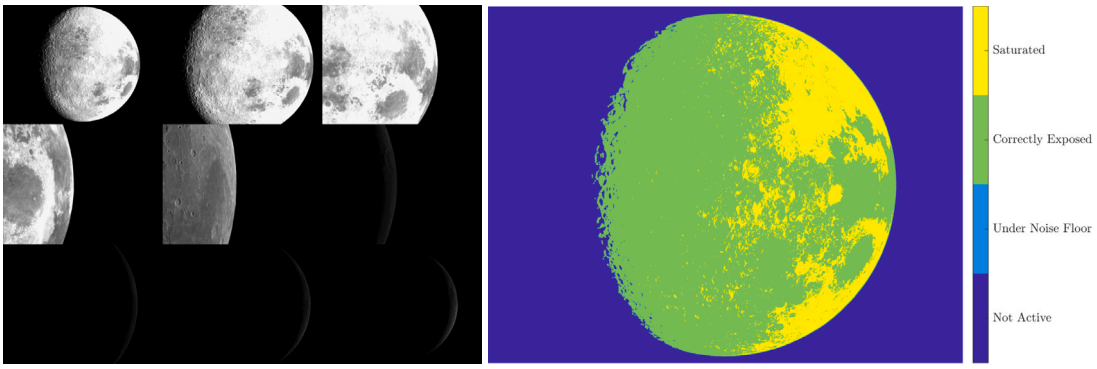
Even when the body is well known, the choice of an exposure time delivering good-exposure images could be problematic, mainly due to approximate modeling of the target body brightness variations along the imaging trajectory. In fact, even if the overall albedo is known, regional variations or specific relative camera and/or Sun geometry conditions could cause surges or drops of brightness in the image. For the Earth certification pass of Artemis I, the preset camera exposure resulted in very over-exposed images due to the bright clouds and Antarctic ice sheet [6]. During the Moon flyby, the same happened due to the large phase angle excursion and the non-linearity of the brightness behavior at close distances from the surface. A simplified rendering model was put in place to determine exposure time brackets on the ≈ 0.1 ms order of magnitude as a function of distance from the lit limb to the terminator [5].

The tuning of the exposure time could be aided by the usage of a physically-based, validated render engine. Let us focus on the same trajectory of Section 6.2, restricted to the frames generated during the periselenium passage, shown in Fig. 40(a). This passage is characterized by a large excursion angle of the phase angle spanning from about 40 deg to 150 deg. For each frame, the image can be segmented based on the amount of pixel illumination with respect to the noise and saturation thresholds. An example for the first frame is shown in Fig. 40(b).

The *optimal* exposure time range is now defined as the one that allows for at least $X\%$ of the active pixels above the noise threshold and at maximum $1-X\%$ of the active pixels under the saturation level. Values outside these boundaries are considered a loss of information. Fig. 41 reports the trend of the information loss along the entire trajectory for different exposure times. It is worth noting that this is equivalent to plotting the empirical Cumulative Distribution Function (CDF) for the *time before saturation* and the Complementary CDF (CCDF) for the *time before noise* for each active pixel. These two quantities are explicitly written in Eqs. (61) and (62) respectively, being Dynamic Range (DNR) the ratio in dB of the signal at saturation versus the minimum detectable signal.

$$\Delta t_{sat} = \frac{\text{FWC}}{S} \tag{61}$$

$$\Delta t_{noise} = \frac{\text{FWC}/(10^{\text{DNR}/20})}{S} \tag{62}$$



(a) Example images along the trajectory (b) Exposure segmentation for the first image

Fig. 40. Segmentation of exposure time quality during a Moon flyby with large phase angle excursion. A pixel is considered well-exposed when its value is under the saturation level and over the noise threshold.

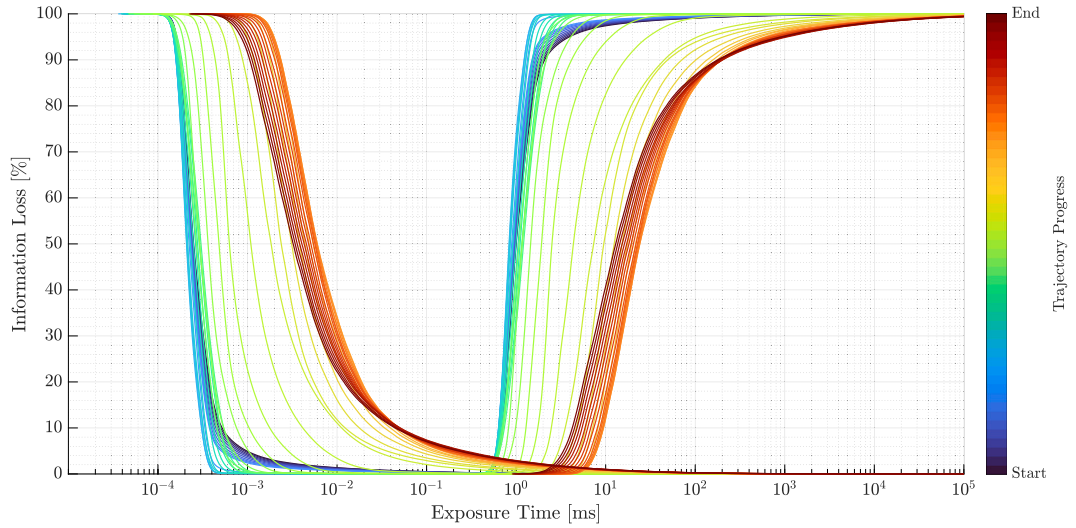


Fig. 41. Tuning of optimal exposure time brackets along a Moon flyby trajectory. The information can be lost either because of over-saturation or under-noise pixels. The sweet spot lies in the middle of these two regions. The optimal exposure time range shifts as the trajectory progresses in time.

The optimal exposure time range lies in between the two curves. As the trajectory progresses and the phase angle increases, both curves shift to the right, indicating, as expected, that a longer exposure time is required to maintain a well-exposed image.

Using a $X = 10\%$ threshold, the optimal exposure time range for the scenario and camera considered lies between 0.2 ms and 0.6 ms. If the threshold were lower, the boundaries of the optimal exposure time range would narrow to ensure the desired level of information retrieval throughout the entire trajectory.

The picture planning team is now able to refine one or more exposure time brackets at the different mission phases. Alternatively, the camera design parameters that are linked with the light collection capabilities (e.g., A_{cam}, μ, QE, T, etc...) could be tuned to find the desired exposure time design space.

7. Conclusions

The article presented a physically-based rendering technique for the generation of images of celestial bodies with radiometric and geometric consistency. Each pixel of the image can be linked with clear physical and radiometric quantities. The methodology has been thoroughly validated against multiple tests, using both analytical laws and real images. The rendering performance is competitive with respect to existing available tools. The proposed methodology was shown to be useful for a variety of applications, such as the use of HIL optical stimulators or the tuning of optimal exposure time brackets.

In this sense, this article aimed to present a flexible and efficient solution, particularly well-suited for scenarios where the complexity of more computationally demanding techniques is unnecessary. Rather than positioning itself as a one-size-fits-all solution, it offers a valuable and practical alternative, especially at mid-to-far ranges, where the influence of secondary surface illumination is considerably reduced, and more complex ray-tracing techniques might not be required.

In particular, the rendering technique supports a variety of applications, including optical navigation experiments involving planetary limbs, feature tracking of craters and other surface elements, and general radiometric studies for camera system design and optimization.

CRediT authorship contribution statement

Andrea Pizzetti: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Paolo Panicucci:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Francesco Capolupo:** Writing – review & editing, Project administration. **Francesco Topputo:** Supervision, Project administration.

Funding sources

The PhD scholarship of A.P. is sponsored by The European Union under the NextGenerationEU (NGEU) programme (Mission 4, Component 1, CUP \D43C23002190008). This work has been conducted under ESA, Italy Contract No. 4000139932/22/NL/CRS within the General Support Technology Programme (GSTP) through the support of the national delegation of Italy (ASI). P.P. and F.T. are sponsored by EXTREMA, a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant Agreement No. 864697).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to acknowledge the maintainers of ESA's Planetary Science Archive (PSA) and NASA's Planetary Data System (PDS) and the data curators for the SMART-1 and Dawn missions for generating the database and making available the images. The authors would also like to acknowledge Fabio Ornati for helping with the usage of the optical stimulator and for providing the ground pictures of the Moon. The authors would like to acknowledge the users of the ABRAM tool who gave constructive feedback and helped in its development.

References

[1] M. Kisantal, S. Sharma, T.H. Park, D. Izzo, M. Märtens, S. D'Amico, Satellite pose estimation challenge: Dataset, competition design, and results, IEEE Trans. Aerosp. Electron. Syst. 56 (5) (2020) 4083–4098, http://dx.doi.org/10.1109/taes.2020.2989063, arXiv:1911.02050.

[2] T.H. Park, S. D'Amico, Bridging the domain gap in satellite pose estimation, IEEE Trans. Aerosp. Electron. Syst. 59 (1) (2023) 123–134.

[3] S. D'Amico, J.-S. Ardaens, G. Gaias, H. Benninghoff, B. Schlepp, J. Jørgensen, Noncooperative rendezvous using angles-only optical navigation: system design and flight results, J. Guid. Control Dyn. 36 (6) (2013) 1576–1595, http://dx.doi.org/10.2514/1.59236.

[4] P. Panicucci, C. Balossi, F. Ornati, F. Piccolo, A. Pizzetti, F. Topputo, F. Capolupo, et al., What if star trackers were navigation cameras? in: 35th AAS/AIAA Space Flight Mechanics Meeting, 2025, pp. 1–23.

[5] K.R. Kobylka, Camera exposure time determination for artemis I lunar flyby, in: 4th Space Imaging Workshop, 2024.

[6] R. Inman, G. Holt, J. Christian, K.W. Smith, C. D'Souza, Artemis I optical navigation system performance, in: AIAA SCITECH 2024 Forum, 2024, p. 0514.

[7] J. Lebreton, R. Brochard, M. Baudry, G. Jonniaux, A. Hadj Salah, K. Kanani, M. Le Goff, A. Masson, N. Ollagnier, P. Panicucci, A. Proag, C. Robin, Image simulation for space applications with the SurRender software, 2021, arXiv preprint arXiv:2106.11322. arXiv:2106.11322. URL: https://arxiv.org/abs/2106.11322.

[8] R. Brochard, J. Lebreton, C. Robin, K. Kanani, G. Jonniaux, A. Masson, N. Despré, A. Berjaoui, Scientific image rendering for space scenes with the SurRender software, 2018, arXiv preprint arXiv:1810.01423.

[9] N. Rowell, S. Parkes, M. Dunstan, O. Dubois-Matra, PANGU: Virtual spacecraft image generation, in: 5th Int. Conf. on Astrodynamics Tools and Techniques, ICATT, 2012.

[10] I. Martin, M. Dunstan, M. Gestido, Planetary surface image generation for testing future space missions with PANGU, in: Proceedings of the 2nd RPI Space Imaging Workshop, Saratoga Springs, NY, USA, 2019.

[11] I. Martin, M. Dunstan, M. Sanchez-Gestido, J. Belhadj, Real-time event-based sensor simulation for space applications, in: 4th Space Imaging Workshop, 2024.

[12] C. Aiazzi, A. Jain, A. Gaut, A. Young, A. Elmquist, Iris: High-fidelity perception sensor modeling for closed-loop planetary simulations, in: AIAA SCITECH 2022 Forum, 2022, p. 1433, http://dx.doi.org/10.2514/6.2022-1433.

[13] R.T. Eapen, R.R. Bhaskara, M. Majji, Narpa: Navigation and rendering pipeline for astronautics, 2022, arXiv preprint arXiv:2211.01566.

[14] J. Villa, J. Mcmahon, I. Nesnas, Image rendering and terrain generation of planetary surfaces using source-available tools, in: Proceedings of the 46th Annual AAS Guidance, Navigation & Control Conference, Breckenridge, CO, USA, 2023, pp. 1–24.

[15] M. Pajusalu, I. Iakubivskyi, G.J. Schwarzkopf, O. Knuuttila, T. Väisänen, M. Bührer, M.F. Palos, H. Teras, G. Le Bonhomme, J. Praks, et al., SISPO: space imaging simulator for proximity operations, PLoS One 17 (3) (2022) e0263882, http://dx.doi.org/10.1371/journal.pone.0263882, arXiv:2105.06771.

[16] L. Bingham, J. Kincaid, B. Weno, N. Davis, E. Paddock, C. Foreman, Digital lunar exploration sites unreal simulation tool (dust), in: 2023 IEEE Aerospace Conference, IEEE, 2023, pp. 1–12, http://dx.doi.org/10.1109/aero55745.2023.10115607.

[17] M. Pugliatti, C. Buonagura, D. Pisanti, N. Faraco, A. Pizzetti, M. Maestrini, F. Topputo, et al., Design and cases studies of CORTO, an open access rendering tool for celestial and artificial bodies, in: 75th International Astronautical Congress, IAC 2024, 2024, pp. 1–10, http://dx.doi.org/10.52202/078357-0122.

[18] R. Bhaskara, G. Georgakis, J. Nash, M. Cameron, J. Bowkett, A. Ansar, M. Majji, P. Backes, Icy moon surface simulation and stereo depth estimation for sampling autonomy, 2024, http://dx.doi.org/10.1109/aero58975.2024.10521439, arXiv preprint arXiv:2401.12414. arXiv:2401.12414.

[19] G. Rufino, A. Moccia, Laboratory test system for performance evaluation of advanced star sensors, J. Guid. Control Dyn. 25 (2) (2002) 200–208, http://dx.doi.org/10.2514/2.4888.

[20] B.G. Boone, J.R. Bruzzi, W.F. Dellinger, B.E. Kluga, K.M. Strobehn, Optical simulator and testbed for spacecraft star tracker development, in: M.A. Kahan (Ed.), Optical Modeling and Performance Predictions II, SPIE, San Diego, CA, 2005, http://dx.doi.org/10.1117/12.619133.

[21] J.A. Tappe, Development of Star Tracker System for Accurate Estimation of Spacecraft Attitude (Ph.D. thesis), Naval Postgraduate School, Monterey, CA, 2009.

[22] M.A. Samaan, S.R. Steffes, S. Theil, Star tracker real-time hardware in the loop testing using optical star simulator, Spacefl. Mech. 140 (2011).

[23] D. Rössler, D.A.K. Pedersen, M. Benn, J.L. Jørgensen, Optical stimulator for vision-based sensors, Adv. Opt. Technol. 3 (2014) 199–207, http://dx.doi.org/10.1515/aot-2013-0045.

[24] C. Beierle, S. D'Amico, Variable-magnification optical stimulator for training and validation of spaceborne vision-based navigation, J. Spacecr. Rockets 56 (4) (2019) 1060–1072, http://dx.doi.org/10.2514/1.a34337.

[25] P. Panicucci, F. Topputo, The tinyv3rse hardware-in-the-loop vision-based navigation facility, Sensors 22 (23) (2022) 9333, http://dx.doi.org/10.3390/s22239333.

[26] F. Ornati, P. Panicucci, E. Andreis, F. Topputo, et al., RETINA: a highly-versatile optical facility for camera-in-the-loop testing of spaceborne vision-based sensors, in: 46th AAS Guidance, Navigation and Control Conference, 2024, pp. 1–19.

[27] P. Tsiotras, ASTROS: A 5DOF experimental facility for research in space proximity operations, Adv. Astronaut. Sci. 151 (2014) 717–730.

[28] R. Zappulla, J. Virgili-Llop, C. Zagaris, H. Park, M. Romano, Dynamic air-bearing hardware-in-the-loop testbed to experimentally evaluate autonomous spacecraft proximity maneuvers, J. Spacecr. Rockets 54 (4) (2017) 825–839, http://dx.doi.org/10.2514/1.A33769.

[29] M. Dor, P. Tsiotras, ORB-SLAM applied to spacecraft non-cooperative rendezvous, in: 2018 Space Flight Mechanics Meeting, 2018, http://dx.doi.org/10.2514/6.2018-1963.

[30] Y.K. Nakka, R.C. Foust, E.S. Lupu, D.B. Elliott, I.S. Crowell, S.-J. Chung, F.Y. Hadaegh, A six degree-of-freedom spacecraft dynamics simulator for formation control research, in: 2018 AAS/AIAA Astrodynamics Specialist Conference, Snowbird, UT, 2018.

[31] T. Boge, T. Rupp, K. Landzettel, T. Wimmer, C. Mietner, J. Bosse, B. Thaler, Hardware in the loop simulator für renezvous und docking manöver, in: Deutsche Luft- Und Raumfahrtkongress 2009, 2009.

[32] L. Pasqualetto Cassinis, A. Menicucci, E. Gill, I. Ahrns, J. Gil Fernandez, On-ground validation of a CNN-based monocular pose estimation system for uncooperative spacecraft, in: 8th European Conference on Space Debris, 8, 2021.

[33] T.H. Park, J. Bosse, S. D'Amico, Robotic testbed for rendezvous and optical navigation: Multi-source calibration and machine learning use cases, 2021, arXiv preprint arXiv:2108.05529. arXiv:2108.05529.

[34] H. Krüger, S. Theil, TRON-hardware-in-the-loop test facility for lunar descent and landing optical navigation, IFAC Proc. Vol. 43 (15) (2010) 265–270, http://dx.doi.org/10.3182/20100906-5-jp-2022.00046.

[35] A. Derobertis, T. Kuwahara, P. Lunghi, Vision-based positioning via linked neural networks in lunar environment, in: Proceedings of the 75th International Astronautical Congress, IAC 2024, Milan, Italy, 2024, pp. 1–15.

[36] M. Pugliatti, C. Buonagura, F. Topputo, Corto: The celestial object rendering tool at dart lab, Sensors 23 (23) (2023) 9595, http://dx.doi.org/10.3390/s23239595.

[37] J.T. Kajiya, The rendering equation, SIGGRAPH Comput. Graph. 20 (4) (1986) 143–150, http://dx.doi.org/10.1145/15886.15902.

[38] U.S.N.B. of Standards, F.E. Nicodemus, Geometrical Considerations and Nomenclature for Reflectance, vol. 160, US Department of Commerce, National Bureau of Standards Washington, DC, USA, 1977, http://dx.doi.org/10.6028/nbs.mono.160.

[39] J.M. Palmer, B.G. Grant, The Art of Radiometry, SPIE, 2009, http://dx.doi.org/10.1117/3.798237.

[40] J.D. Foley, Computer Graphics: Principles and Practice, vol. 12110, Addison-Wesley Professional, 1996.

[41] T. Whitted, An improved illumination model for shaded display, in: ACM Siggraph 2005 Courses, 2005, pp. 4–es, http://dx.doi.org/10.1145/1198555.1198743.
[42] A. Appel, Some techniques for shading machine renderings of solids, in: Proceedings of the Spring Joint Computer Conference, AFIPS, 1968, pp. 37–45, http://dx.doi.org/10.1145/1468075.1468082.
[43] R.L. Cook, T. Porter, L. Carpenter, Distributed ray tracing, in: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques, 1984, pp. 137–145, http://dx.doi.org/10.1145/280811.280978.
[44] J.T. Kajiya, The rendering equation, in: Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, 1986, pp. 143–150, http://dx.doi.org/10.1145/280811.280987.
[45] J. Pineda, A parallel algorithm for polygon rasterization, in: Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques, 1988, pp. 17–20, http://dx.doi.org/10.1145/378456.378457.
[46] E.E. Catmull, A Subdivision Algorithm for Computer Display of Curved Surfaces, The University of Utah, 1974.
[47] B.J. Buratti, J.A. Mosher, T.V. Johnson, Albedo and color maps of the saturnian satellites, Icarus 87 (2) (1990) 339–357, http://dx.doi.org/10.1016/0019-1035(90)90138-y.
[48] T. Johnson, L. Soderblom, J. Mosher, G. Danielson, A. Cook, P. Kupferman, Global multispectral mosaics of the icy Galilean satellites, J. Geophys. Res.: Solid Earth 88 (B7) (1983) 5789–5805, http://dx.doi.org/10.1029/jb088ib07p05789.
[49] T. Roatsch, E. Kersten, K.-D. Matz, F. Preusker, F. Scholten, R. Jaumann, C.A. Raymond, C.T. Russell, High resolution vesta high altitude mapping orbit (HAMO) atlas derived from dawn framing camera images, Planet. Space Sci. 73 (1) (2012) 283–286, http://dx.doi.org/10.1016/j.pss.2012.08.021.
[50] M.T. Bland, T.L. Becker, K.L. Edmundson, T. Roatsch, B.A. Archinal, D. Takir, G.W. Patterson, G.C. Collins, P.M. Schenk, R.T. Pappalardo, et al., A new Enceladus global control network, image mosaic, and updated pointing kernels from Cassini's 13-year mission, Earth Space Sci. 5 (10) (2018) 604–621, http://dx.doi.org/10.1029/2018ea000399.
[51] H. Sato, M. Robinson, B. Hapke, B. Denevi, A. Boyd, Resolved Hapke parameter maps of the moon, J. Geophys. Res.: Planets 119 (8) (2014) 1775–1805, http://dx.doi.org/10.1002/2013je004580.
[52] T.P. Lester, M.L. McCall, J. Tatum, Theory of planetary photometry, J. R. Astron. Soc. Can. 73 (1979) 233–257, vol. 73, Oct. 1979, p. 233-257.
[53] A. Pizzetti, Normal texture maps of planets and moons of the solar system, 2025, http://dx.doi.org/10.5281/zenodo.14936972.
[54] R. Wahl, M. Massing, P. Degener, M. Guthe, R. Klein, Scalable compression and rendering of textured terrain data, in: International Conference in Central Europe on Computer Graphics and Visualization, 2004.
[55] F. Losasso, H. Hoppe, Geometry clipmaps: terrain rendering using nested regular grids, in: ACM Siggraph 2004 Papers, 2004, pp. 769–776, http://dx.doi.org/10.1145/1186562.1015799.
[56] S. Myint, A. Jain, J. Cameron, C. Lim, Large terrain modeling and visualization for planets, in: 2011 IEEE Fourth International Conference on Space Mission Challenges for Information Technology, IEEE, 2011, pp. 177–183, http://dx.doi.org/10.1109/smc-it.2011.11.
[57] J.C. Hart, Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces, Vis. Comput. 12 (10) (1996) 527–545, http://dx.doi.org/10.1007/s003710050084.
[58] M. Rost, B. Licea-Kane, OpenGL Shading Language (4th Edition), fourth ed., Addison-Wesley Professional, 2016, URL: https://www.oreilly.com/library/view/opengl-shading-language/9780134425817/.
[59] M. Vincendon, Y. Langevin, F. Poulet, J.-P. Bibring, B. Gondet, Retrieval of surface lambert albedos and aerosols optical depths using OMEGA near-IR EPF observations of mars, in: 38th Annual Lunar and Planetary Science Conference, (vol. 1338) 2007, p. 1650.
[60] U. Dyudina, X. Zhang, L. Li, P. Kopparla, A.P. Ingersoll, L. Dones, A. Verbiscer, Y.L. Yung, Reflected light curves, spherical and bond albedos of Jupiter-and Saturn-like exoplanets, Astrophys. J. 822 (2) (2016) 76, http://dx.doi.org/10.3847/0004-637x/822/2/76.
[61] K. Muinonen, J. Torppa, X.-B. Wang, A. Cellino, A. Penttilä, Asteroid lightcurve inversion with Bayesian inference, Astron. Astrophys. 642 (2020) A138, http://dx.doi.org/10.1051/0004-6361/202038036.
[62] B. Hapke, Bidirectional reflectance spectroscopy: 1. Theory, J. Geophys. Res.: Solid Earth 86 (B4) (1981) 3039–3054, http://dx.doi.org/10.1029/jb086ib04p03039.
[63] B. Hapke, Bidirectional reflectance spectroscopy: 6. Effects of porosity, Icarus 195 (2) (2008) 918–926, http://dx.doi.org/10.1016/j.icarus.2008.01.003.
[64] M. Oren, S.K. Nayar, Generalization of Lambert's reflectance model, in: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, 1994, http://dx.doi.org/10.1145/192161.192213, URL: https://api.semanticscholar.org/CorpusID:122480.
[65] C.B. Duane, Close-range camera calibration, Photogramm. Eng. 37 (8) (1971) 855–866.
[66] D. Brown, Decentering distortion of lenses, Photogramm. Eng. 32 (3) (1996) 444–462.
[67] S. Krishnan, C. Baru, C. Crosby, Evaluation of MapReduce for gridding LIDAR data, in: 2010 IEEE Second International Conference on Cloud Computing Technology and Science, IEEE, 2010, pp. 33–40, http://dx.doi.org/10.1109/cloudcom.2010.34.
[68] L. Feng, Q. Li, K. Chen, Y. Li, X. Tong, X. Wang, H. Lu, Y. Li, The gridding method for image reconstruction of nonuniform aperture synthesis radiometers, IEEE Geosci. Remote. Sens. Lett. 12 (2) (2014) 274–278, http://dx.doi.org/10.1109/lgrs.2014.2335413.
[69] R.L. Cook, Stochastic sampling in computer graphics, ACM Trans. Graph. 5 (1) (1986) 51–72, http://dx.doi.org/10.1145/7529.8927.
[70] R.D. Fiete, B.D. Paul, Modeling the optical transfer function in the imaging chain, Opt. Eng., Bellingham 53 (8) (2014) 083103, http://dx.doi.org/10.1117/1.OE.53.8.083103.
[71] R.D. Gow, D. Renshaw, K. Findlater, L. Grant, S.J. McLeod, J. Hart, R.L. Nicol, A comprehensive tool for modeling CMOS image-sensor-noise performance, IEEE Trans. Electron Devices 54 (6) (2007) 1321–1329, http://dx.doi.org/10.1109/ted.2007.896718.
[72] H. Wach, E.R. Dowski Jr., Noise modeling for design and simulation of computational imaging systems, in: Visual Information Processing XIII, vol. 5438, SPIE, 2004, pp. 159–170.
[73] P. Fiorentin, P. Iacomussi, G. Rossi, Characterization and calibration of a CCD detector for light engineering, IEEE Trans. Instrum. Meas. 54 (1) (2005) 171–177, http://dx.doi.org/10.1109/tim.2004.834055.
[74] P.R. Ailuri, O. Skorka, N. Li, R. Ispasoiu, V. Korobov, Correlation of photoresponse blooming metrics with image quality in CMOS image sensors for mobile photography, Electron. Imaging 28 (2016) 1–6, http://dx.doi.org/10.2352/issn.2470-1173.2016.13.iqsp-203.
[75] K.W. Benoist, R.H. Schleijpen, Modeling of the over-exposed pixel area of CCD cameras caused by laser dazzling, in: Technologies for Optical Countermeasures XI; and High-Power Lasers 2014: Technology and Systems, vol. 9251, SPIE, 2014, pp. 105–113, http://dx.doi.org/10.1117/12.2066305.
[76] B. Foing, G.D. Racca, A. Marini, E. Evrard, L. Stagnaro, M. Almeida, D. Koschny, D. Frew, J. Zender, J. Heather, et al., SMART-1 mission to the moon: status, first results and goals, Adv. Space Res. 37 (1) (2006) 6–13, http://dx.doi.org/10.1016/j.asr.2005.12.016.
[77] J.-L. Josset, S. Beauvivre, P. Cerroni, M.C. De Sanctis, P. Pinet, S. Chevrel, Y. Langevin, M.A. Barucci, P. Plancke, D. Koschny, et al., Science objectives and first results from the SMART-1/AMIE multicolour micro-camera, Adv. Space Res. 37 (1) (2006) 14–20, http://dx.doi.org/10.1016/j.asr.2005.06.078.
[78] B. Grieger, The Calibration of AMIE Images, Technical Report, ESA Technical Note S1-AMIE-SGSTN-013, 2008.
[79] I. Belgacem, G. Jonniaux, F. Schmidt, Image processing for precise geometry determination, Planet. Space Sci. 193 (2020) 105081, http://dx.doi.org/10.1016/j.pss.2020.105081, arXiv:2009.02159.
[80] D. Despan, S. Erard, A. Barucci, J.-L. Josset, S. Beauvivre, S. Chevrel, D. Koschny, M. Almeida, A. Team, et al., Geometrical analysis of AMIE/Smart-1 images and applications to photometric studies of the lunar surface, in: AAS/Division for Planetary Sciences Meeting Abstracts# 38, vol. 38, 2006, pp. 57–09.
[81] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612, http://dx.doi.org/10.1109/TIP.2003.819861.
[82] M.J. Kidger, Fundamental Optical Design, SPIE Optical Engineering Press, 2001.
[83] D.E. Lee, Gateway destination orbit model: A continuous 15 year NRHO reference trajectory, NASA Johns. Space Cent. White Pap. (2019).
[84] A. Pizzetti, P. Panicucci, F. Topputo, A radiometric consistent render procedure for planets and moons, in: 4th Space Imaging Workshop, 2024, pp. 1–3.
[85] P. Panicucci, F. Ornati, F. Topputo, Design of a low-aberration variable-magnification optical stimulator for vision system hardware-in-the-loop testing, IEEE Trans. Aerosp. Electron. Syst. 61 (6) (2025) 15224–15243, http://dx.doi.org/10.1109/TAES.2025.3580393.
[86] F. Ornati, P. Panicucci, A. Pizzetti, F. Topputo, A radiometric calibration procedure for optical hardware-in-the-loop stimulators, in: 4th Space Imaging Workshop, 2024.
[87] F. Ornati, P. Panicucci, A. Pizzetti, F. Capolupo, F. Topputo, On the radiometric calibration of optical hardware-in-the-loop stimulators, J. Spacecr. Rockets (2026).
[88] W.M. Owen, P.J. Dumont, C.D. Jackman, Optical navigation preparations for new horizons pluto flyby, in: International Symposium on Space Flight Dynamics, 2012.