Leveraging a mission Digital Twin technology for autonomous lunar rover navigation and mission validation

Luca Andolfi[(1)], **Giuseppe Tomasicchio**[(2)], **Simone Giannattasio**[(3)], **Salvatore Cassano**[(4)], **Luca Ostrogovich**[(5)], **Alfredo Renga**[(6)], **Nicolò Galletta**[(7)], **Michèle Lavagna**[(8)]

[(1)(2)(3)]Telespazio, Italy, Rome, 00156, +39 0640795028 {luca.andolfi, giuseppe.tomasicchio, simone.giannattasio}@telespazio.com
[(4)]Department of Mechanical and Aerospace Engineering, University of Rome La Sapienza, Rome, Italy, cassano.1886912@studenti.uniroma1.it
[(5)(6)]Department of Industrial Engineering, University of Naples Federico II, Naples, Italy, {luca.ostrogovich, alfredo.renga}@unina.it
[(7)(8)]Aerospace Science and Technology Department, Politecnico di Milano, Milan, Italy, nicolo.galletta@mail.polimi.it, michelle.lavagna@polimi.it

Abstract

In recent years, the renewed interest in lunar surface exploration has highlighted the need for accurate and reliable navigation techniques for the users, included long-range rovers operating in unstructured environments. In this context, the objective of this study is to evaluate a Multi-Sensor Fusion Algorithm that includes Visual-Based Navigation (VBN) pipeline integrating stereo vision, inertial data, and wheel odometry through a feedforward Extended Kalman Filter (EKF) architecture. The proposed system is tested within a high-fidelity lunar surface simulation environment, the Telespazio Interactive Mission Modeling, Visualization & Validation tool (IMMV^2) developed within the eXtended Reality Lab (XR-Lab). The simulation leverages different modules of the tool, especially the Visual Scenario Generator (VSG) module of the IMMV^2 to generate photorealistic synthetic datasets based on the Lunar Reconnaissance Orbiter's (LRO) Lunar Orbiter Laser Altimeter's (LOLA) Digital Elevation Model (DEM) augmented with fractal noise and procedurally generated rocks and craters. Results over a 500-meter reference trajectory demonstrate that the EKF integration significantly stabilizes pose estimation, yielding a final position error of approximately 7.5 meters (1.5% of the travelled distance). Additionally, the simulation environment supports the creation of automatically labelled synthetic images for the training of neural networks to be used for hazard and loop detection modules.

Keywords: Interactive Mission Modeling Visualization & Validation Tool, Mission Digital Twin, Lunar Guidance & Navigation, Visual Navigation Techniques, Mission & Trajectory Design.

Acronyms

AWGN: Additive White Gaussian Noise

BoW: Bag of Words

DEM: Digital Elevation Model

DUST: Digital lunar exploration sites Unreal Simulation Tool

EKF: Extended Kalman Filter

ESA: European Space Agency

XR-Lab: eXtended Reality Laboratory

FAST: Features from Accelerated Segment Test

IMMV^2: Interactive Mission Modeling, Visualization & Validation tool

IMU: Inertial Measurement Unit

ISRU: In-Situ Resource Utilization

Company General Use

LK: Lucas–Kanade

JAXA: Japan Aerospace Exploration Agency

LOLA: Lunar Orbiter Laser Altimeter

LRO: Lunar Reconnaissance Orbiter

LUPEX: LUnar Polar EXploration

MLESAC: Maximum Likelihood Estimator Sample Consensus

NASA: National Aeronautics and Space Administration

ORB: Oriented FAST and Rotated BRIEF

PnP: Perspective-n-Points

RANSAC: RANdom SAmple Consensus

SGM: Semi-Global Matching

UE5: Unreal Engine 5

UDP: User Datagram Protocol

VBN: Visual-Based Navigation

VO: Visual Odometry

VSG: Visual Scenario Generator

WO: Wheel Odometry

XR-Lab: eXtended Reality Lab

1. Introduction

Space Agencies current exploration roadmaps, included NASA's Artemis programme [1] and ESA's Explore 2040 roadmap [2], highlight the Moon as a key platform for validating technologies related to long-term habitation, in-situ resource utilization (ISRU), and autonomous robotic operations is extraterrestrial environments. In parallel, the use of advanced visualization and rendering technologies such as NASA's Digital Lunar Exploration Sites Unreal Simulation Tool (DUST) is transforming how future space missions are conceived, modelled and tested [3].

Recent missions such as Yutu-2 and Chandrayaan-3 have demonstrated the critical role of surface rovers in autonomous exploration and scientific operations [4] [5]. Upcoming missions, including ESA's Argonaut lander and JAXA's LUPEX rover, further stress the need for long-range mobility, extended operational lifetimes, and robust hazard avoidance, particularly in the complex and poorly illuminated terrain near the lunar South Pole [6] [7]. In such operational contexts, achieving reliable and drift-resilient pose estimation is essential, requiring robust visual navigation and multi-sensor fusion architectures that combine stereo images, inertial measurements, and, when available, range-based observables. Validating such pipelines in realistic terrain, lighting, and dynamic conditions is a necessary step before deployment on real hardware.

This study focuses on the validation of a multi-sensor fusion approach that compares two Visual-Based Navigation (VBN) algorithms, integrating Inertial Measurement Unit (IMU) and Wheel Odometry (WO) observables in a framework based on an Extended Kalman Filter (EKF). Synthetic stereo images, inertial data, and odometry measurements are generated within a high-fidelity digital environment and used to assess the system's performance along a 500-meter traverse on a simulated lunar south polar surface. The EKF effectively fuses the multi-modal measurements, improving localization accuracy and reducing drift compared to standalone Visual Odometry (VO).

The simulation environment is developed using the Interactive Mission Modeling, Visualization & Validation tool (IMMV^2) [8], [9], [10] which uses augmented Lunar Reconnaissance Orbiter's (LRO) Lunar Orbiter Laser Altimeter's (LOLA) Digital Elevation Models (DEM) [11]. With its functionalities the tool enables End-to-End validation of advanced rover navigation pipelines and the generation of labelled datasets for hazard detection model training.

The paper is structured as follows: Section 1 briefly introduces the context in which the proposed technology is inserted within. Section 2 presents the mission scenario, defines the reference rover trajectory on the lunar south pole surface, then move to a brief description of the IMMV^2. Section 3 describes the modelling of the rover and of its synthetic sensors and presents the two VBN algorithms which are integrated in the sensor-fusion pipeline. Section 4 describes the EKF

filter and its main parameters. Section 5 discusses quantitative results over the reference traverse. Section 6 concludes with lessons learned and states the next targeted improvements to this work.

2. Mission Scenario: Rover Trajectory over the Lunar South Pole

A reference rover trajectory was generated using an A* path planning algorithm, optimizing for minimum travel distance while avoiding terrain regions with slopes exceeding 20°. The resulting nominal path spans approximately 500 meters and features a gradual increase in elevation (135m overall) due to the inherent terrain inclination.

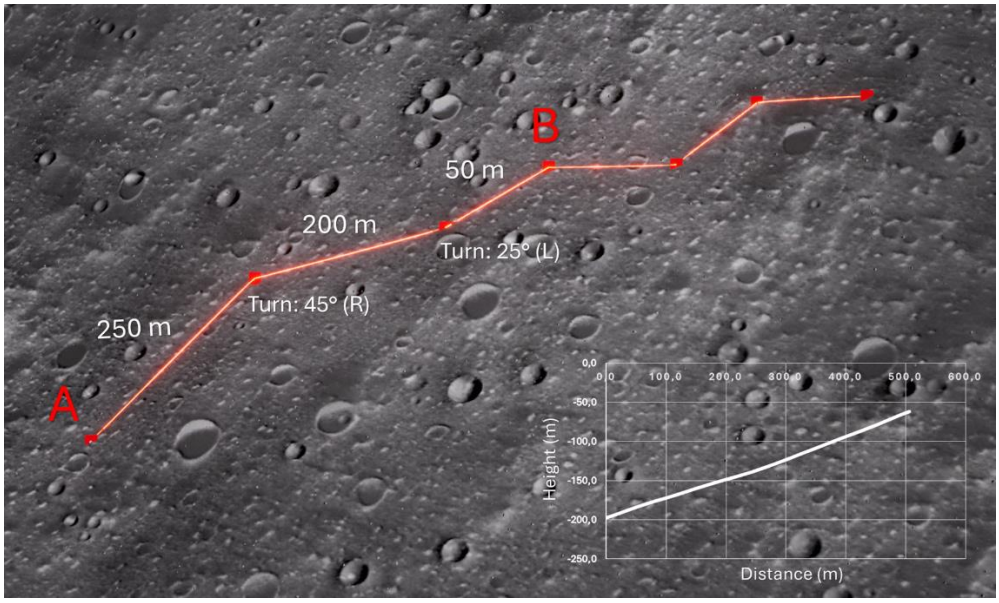

Figure 1: Rover reference path

Figure 1 shows the trajectory on the rendered lunar surface, which is implemented in Unreal Engine as a spline composed of fixed ground control points. The path analysed in this work goes from point A to point B of the spline. It begins with a 250-meter straight segment, followed by a 200-meter rightward arc and a final left turn. The simulated rover follows this path using a proportional control scheme implemented inside the simulator itself, ensuring smooth adherence to the designed route while traversing rocks, craters, and other surface obstacles. The rover follows the trajectory with an average velocity of 1.5 m/s.

This synthetic mission scenario served as the foundation for generating the dataset used throughout this study. The IMMV^2 co-simulation environment enabled precise control over sensor models, camera parameters, and trajectory dynamics, allowing the extraction of high fidelity visual and inertial measurements. These data were used to support both the VBN algorithms.

2.1 The Interactive Mission Modeling, Visualization & Validation Tool - tiles generation and augmentation

The simulated lunar mission scenario considered in this work was developed within the IMMV^2, a digital twin framework designed and developed at Telespazio eXtended Reality Laboratory (XR-Labs) [8]. The IMMV^2 provides a co-simulation environment supporting mission digitalization, realistic 3D modelling, and end-to-end validation through a high-resolution reproduction of the mission. One of the central modules is the VSG, which integrates Unreal Engine 5.1 for high-fidelity rendering and dynamic interaction with external simulation environments via a bilateral User Datagram Protocol (UDP) communication interface. This setup enables real-time “ping-pong” co-simulation, where the virtual environment can react to and influence backend physics-based propagators, fostering the generation of consistent, georeferenced optical and inertial observables across diverse mission profiles.

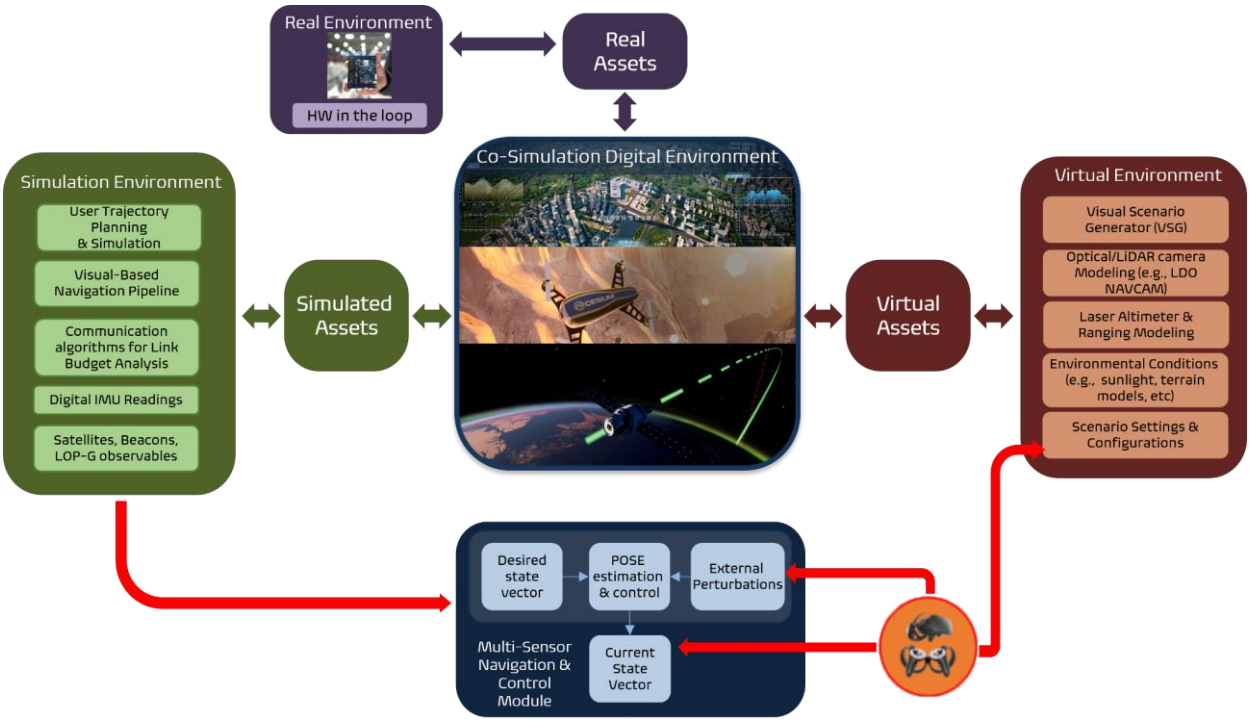

Figure 2: Overall architecture of Telespazio IMMV² tool

In this study, the IMMV² was exploited to simulate a realistic lunar surface scenario centered on the South Pole region, in the proximity of the Shackleton Crater. The reference topography was derived from the 5 m/pixel LRO/LOLA's DEM [11]. To enable rover-scale simulations, this base dataset was further refined to an effective resolution of 10 cm/pixel by integrating fractal noise maps to replicate surface roughness, along with multiple layers of high-resolution texture details to enhance visual realism. Procedural generation techniques were employed to populate the terrain with statistically distributed craters and rocks, resulting in a landscape that exhibits high photorealism and geometric complexity consistent with actual lunar imagery. Craters were generated using statistical distributions governing their position, rim height, radius, and depth. The rocky features consisted of 15 distinct static meshes derived from real lunar samples [12], randomly scaled and oriented according to custom user parameters to accurately represent the target lunar surface site. The final environment incorporates realistic lighting using Unreal Engine 5's Lumen system for ray-traced shadows and reflections, enabling the generation of high resolution synthetic imagery under illumination conditions representative of the lunar South Pole [13], [14], [15]. Furthermore, UE5 Chaos Vehicle physics solver supports multibody dynamics and deformable contact models, allowing rover's suspension behaviour, wheel slippage, and motor torques to influence onboard sensor readings, including those of cameras and IMUs [16].

3. Multi-Sensor Fusion Functional Architecture

To simulate a realistic rover mission scenario, a rover asset and a complete suite of sensors have been modelled within the IMMV². The rover was inspired by Perseverance Rover [17]. The sensors considered are stereo cameras, accelerometers and wheel encoders..

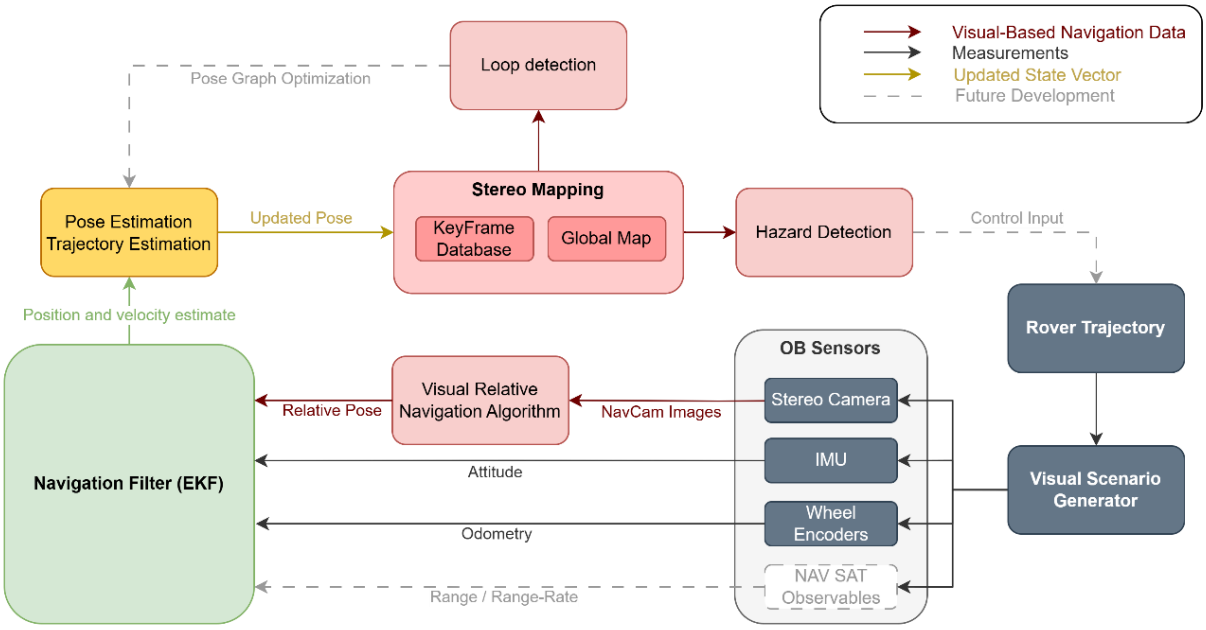

Figure 3: Sensor Fusion Functional Architecture

Figure 3 summarizes the functional architecture of the proposed multi-sensor fusion algorithm. The grey modules represent components within the lunar digital scenario that generate synthetic data, which are processed by the VBN algorithm and fused in the EKF to provide a final estimate of the rover's pose and trajectory. The estimated pose is used by the mapping module to correctly position triangulated 3D points and incrementally build a map of the environment. Keyframes extracted during this process are employed for loop closure detection, which, in the case of a closed trajectory, can significantly reduce drift and improve global consistency. Additionally, these keyframes are processed by a neural network trained for rock detection, enabling the identification of potential hazards along the rover's path. This output will be used in future iterations to generate corrective control inputs for trajectory adjustment and obstacle avoidance.

Several ongoing extensions aim to further enhance the system. A pose graph optimization module is under development to perform trajectory correction after loop closure detection. Among the most significant enhancements currently in progress is the integration of synthetic Navigation Satellites (NAV SATs) observables into the EKF framework. This addition aims to evaluate the potential impact of lunar navigation constellations on ground-based operations and improve the robustness of pose estimation, particularly in visually degraded environments.

3.1 Rover Modelling

The fundamental construction pipeline for a vehicle model in Unreal Engine is based on Blender vehicle rigging, allowing the physical model to be described with different levels of complexity. However, for a basic construction pipeline, a faster and more efficient approach can be sufficient to enable rover mobility in Unreal Engine. Notably, the Chaos Vehicle Physics Engine of Unreal Engine was utilized for this simulation, replacing the deprecated PhysX engine [16].

The process starts with downloading the 3D model of the Perseverance Rover [18] and preprocessing it in Blender to separate the wheels and prepare for rigging. Using the UE4Blender plugin, basic rigging is performed, duplicating the wheel bones to differentiate physical and animated elements. The model is then exported as a Skeletal Mesh (Figure 4).

In Unreal Engine, wheel bones are aligned with the geometry and set as kinematic, allowing the Vehicle Movement Component to manage dynamics. The result is shown in Figure 5. Key parameters are configured in the detail panel, as listed in Table 1, while others can be fine-tuned for added realism.

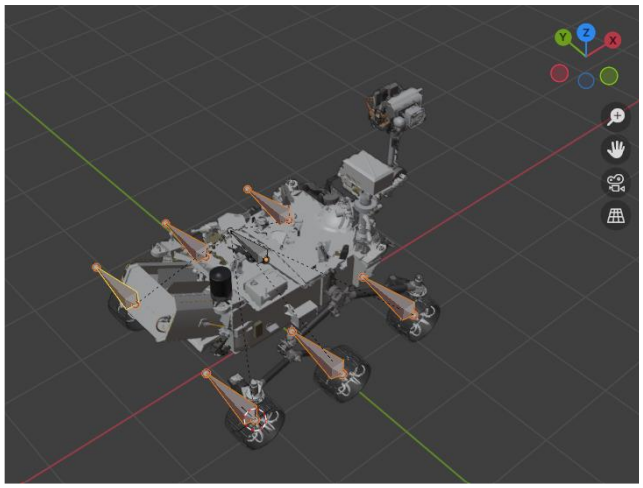

Figure 4: Rover model in Blender

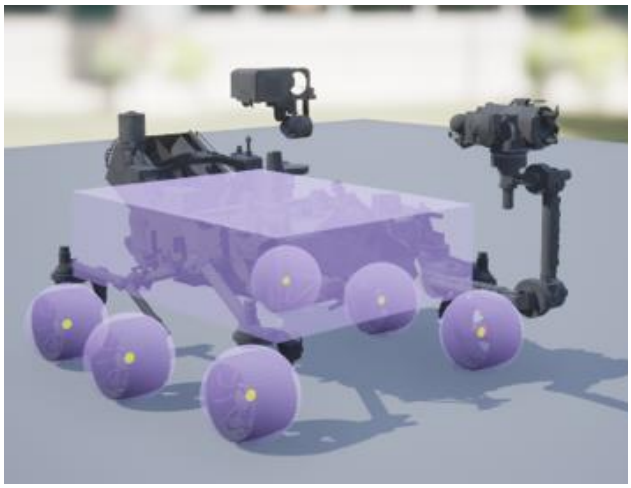

Figure 5: UE Rover model

Parameters	Value
Torque Curve	Assumed Linear
Maximum Torque [Nm]	4500
Maximum RPM	4500
Maximum Steering [deg]	50
Mass [Kg]	300

Table 1: Key parameters of the rover model

3.1.1 Camera Model

A pinhole camera model is assumed for the stereo imaging system. In Unreal Engine, the stereo configuration is implemented using a pair of CineCamera actors rigidly mounted on the simulated rover. The camera parameters are inspired by those of the Perseverance NavCam [19], with modifications to suit the specific simulation requirements, as detailed in Table 2.

Camera Parameter	Value
Camera FOV	96° × 73°
Focal length	19.1 mm
Stereo baseline	42.4 cm
Sensor size	32.77 × 24.58 mm
Image resolution	1378 × 720 px
Left camera position*	[1.0, 0.365, 1.98] m
Right camera position*	[1.0, 0.787, 1.98] m

Table 2: Stereo camera parameters. (*) Positions are given in UE5 left-handed rover frame: x forward, y rightward, z upward.

3.1.2 IMU Model

The rover's inertial measurements are modelled by perturbing ground-truth attitude data extracted from the simulation environment. These true orientations, expressed as Euler angles, are assumed to represent the nominal output of an IMU. To simulate realistic noise characteristics, each attitude measurement is corrupted with additive white Gaussian noise (AWGN), modelling uncertainty in orientation as experienced by space-qualified inertial sensors.

The noisy attitude $\boldsymbol{\vartheta}_{sensor}$ can be represented as:

$$\boldsymbol{\vartheta}_{sensor} = \boldsymbol{\vartheta}_{real} + \aleph(0, \sigma_a^2 \mathbf{I}_3)$$

where $\boldsymbol{\vartheta_{real}}$ denotes the true Euler angles obtained from the simulation, $\aleph(0, \sigma_a^2 \mathbf{I_3})$ is a zero-mean Gaussian noise vector, and $\sigma_a = 0.1°$ is the assumed standard deviation per axis. This represents a conservative estimate based on the performance of typical space-grade IMUs [20].

3.1.3 Encoder Model

To simulate wheel encoders, the angular velocities of all wheels are read directly from Unreal Engine, where they are affected by slippage due to simplified physical interactions between the wheels and the lunar terrain. These angular velocities are then multiplied by the wheel radius to approximate linear displacement, effectively mimicking real encoder readings. The terrain–wheel interactions are modelled using basic physical parameters, including static and dynamic friction coefficients representative of lunar regolith (typically $\mu_s \approx 0.6, \mu_d \approx 0.5$) [21].

The encoder sensor model includes additive white noise and is defined as:

$$\omega_{\text{sensor}} = \omega_{\text{real}} + \aleph(0, \sigma_w)$$

where ω_{real} is the actual angular velocity provided by the simulation, and σ_w is the noise standard deviation. This let us to compute the velocity of the wheel. This setup allows the simulation to realistically capture encoder behaviour under imperfect traction conditions, as would occur during planetary exploration.

3.2 Visual Based Navigation - Relative Navigation

3.2.1 First Approach

This stereo visual odometry pipeline was developed in python to compute a vehicle's 6-DoF trajectory in real time using only synchronized left-right image pairs. The system follows a classic five-stage structure: feature extraction, feature tracking, 3-D reconstruction, pose estimation, and optimisation leveraging tools provided by the well-known Computer Vision python library openCV. The processing begins with Features from Accelerated Segment Test (FAST) corner detection on the left image. The image is splitted in tiles of [10x20] in order to let keypoints (10 best per tile) to be selected all over the frame image. Each keypoint is then tracked to the next frame using a pyramidal Lucas–Kanade (LK) optical-flow implementation with at maximum three levels and a 15x15 pixel window [22]. For every time step, the pipeline also computes a dense disparity map between the rectified stereo pair via Semi-Global Matching (SGM) algorithm [23]. Once disparities are available and the right key-points are computed, the system back-projects all reliable pixels into three-dimensional space using the pinhole stereo model [24]. Eventually an initial rigid transformation $(\mathbf{R}_0, \mathbf{t}_0)$ is obtained with the openCV PnP Solver [25] and to guard against outliers, a RANSAC loop runs one hundred iterations [26]. Finally, an inlier-only least square optimisation minimises the reprojection errors:

$$E = \sum_{i=1}^{N} \|p_i - \pi(\mathbf{R}P_i + \mathbf{t})\|^2$$

Figure 6: First approach VO algorithm

3.2.2 Second Approach

The second visual navigation algorithm is implemented in MATLAB© R2024b, leveraging the Computer Vision Toolbox and built-in support for image and point cloud processing [27]. It is based on the ORB-SLAM2 architecture [28] and follows the pipeline illustrated in Figure 7. The system operates in closed-loop mode using synthetic images captured from the Unreal Engine simulation, though the structure allows future extension to real-time operation via UDP communication.

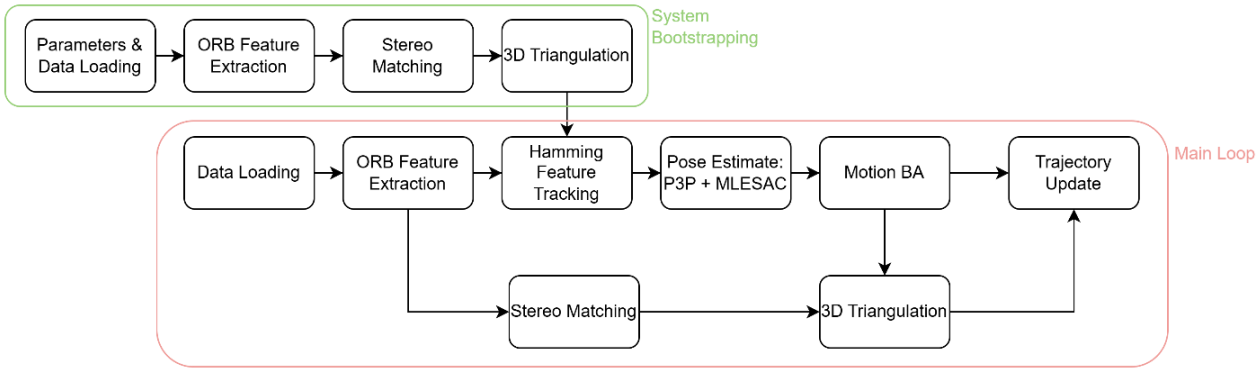

Figure 7: Second approach VO algorithm

During bootstrapping, camera intrinsics and the initial rover pose are obtained from the simulation environment. Also, features extracted from the very first stereo pair are matched and triangulated for the use in the pose estimation of the next frame. After taking the new stereo images pair ORB features are extracted and filtered based on Harris corner response to improve reliability [29], [30]. If more than 7500 features remain, spatial filtering is applied to ensure uniform distribution. Feature tracking between consecutive left camera frames is performed via Hamming distance matching of binary descriptors. Extraction and matching parameters are summarized in Table 3 and Table 4, respectively.

Algorithm	ORB
Number of Levels	8
Scale Factor	1.2

Table 3: Feature extraction parameters

Metric	Hamming Distance
Match Threshold	20
Max Ratio	0.8

Table 4: Feature tracking parameters

Camera pose estimation is performed using MATLAB's *estworldpose* function [31], which implements the Perspective-3-Point algorithm combined with Maximum Likelihood Estimator Sample Consensus (MLESAC) for robust outlier rejection [32], [33]. A confidence level of 99% and a reprojection error threshold of 20 pixels are used to guide inlier selection. Keypoints in the current frame are matched with 3D landmarks previously triangulated from stereo keyframes. The estimated pose is subsequently refined via motion-only bundle adjustment, minimizing the reprojection error of the tracked keypoints to improve local consistency. After pose estimation, the points of the left and right frame are matched and triangulated in the camera reference frame. The pose of the camera allows for the correct positioning of the new triangulated points in the original reference frame.

3.2.3 Mapping

The mapping module is compatible with both the presented VBN algorithms. Depth information is obtained through stereo matching, by extracting binary features from the left and right images and computing disparities between matched pairs. The 3D coordinates of keypoints in the stereo camera frame are then recovered using the stereo camera model:

$$Z = \frac{f_x B}{d}, \qquad X = \frac{(u_l - c_x)Z}{f_y}, \qquad Y = \frac{(v_l - c_y)Z}{f_x}$$

Here, (u_l, v_l) denote the keypoint coordinates in the image, (c_x, c_y) the optical center, d the disparity, (f_x, f_y) the focal lengths in pixels, and B the stereo baseline.

Triangulated points are then transformed into the global frame using the estimated camera poses and stored in a *worldpointset* object [34], which links each point to the corresponding frame stored in an *imageviewset* [35]. Redundant points observed in fewer than two views are culled, while views sharing more than five common points are used for local bundle adjustment.

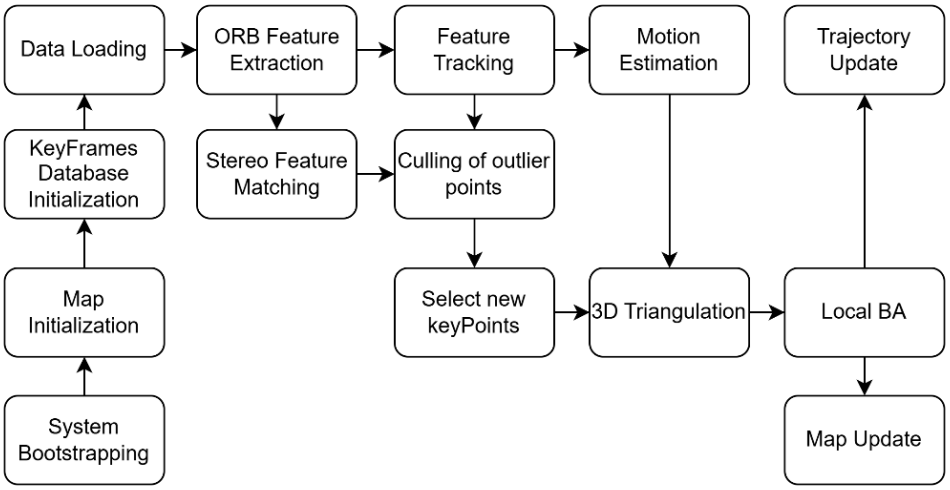

Figure 8: VBN + Mapping pipeline

The resulting point cloud enables a range of downstream applications, including local DEM generation, terrain slope estimation, and cost map creation. integrating additional information from rock and crater detection. These outputs are further enhanced by integrating information from dedicated rock and crater detection modules. In particular, a hazard detection system based on the YOLO11s object detection network [36] automatically identifies rocks in the images and projects the corresponding hazard zones into the reconstructed 3D environment. This capability is essential for all traverses, as it allows for dynamic costmap generation and improves path planning by accounting for obstacles that may not be sufficiently captured through geometric reconstruction alone.

The network outputs bounding boxes corresponding to detected rocks, which are subsequently filtered by size. Those with dimensions exceeding five pixels are retained, and their centres and sizes are stored to define "danger zones". These zones are then used to influence navigation decisions.
The training dataset for the network was generated using the VSG module, which automatically generates lunar surface images with corresponding rock annotations. A total of 3315 images were produced, with 2320 used for training, 663 for validation, and 332 for testing. All images were preprocessed to normalize illumination and remove small rock labels, helping the model focus on more significant hazards. The YOLO network was trained for 10 epochs with a batch size of 9 and an input resolution of 1280 pixels. Performance metrics are summarized in Table 5. As shown in Figure 9, the model demonstrated satisfactory performances also on the dataset of the reference traverse. While a minor decrease in precision and recall was observed, the model remained robust in identifying major obstacles across varying operational conditions.

mAP-50	0.8285
mAP-50-90	0.5311
Box P	0.8455
Box R	0.7134

Table 5: Metrics of the training

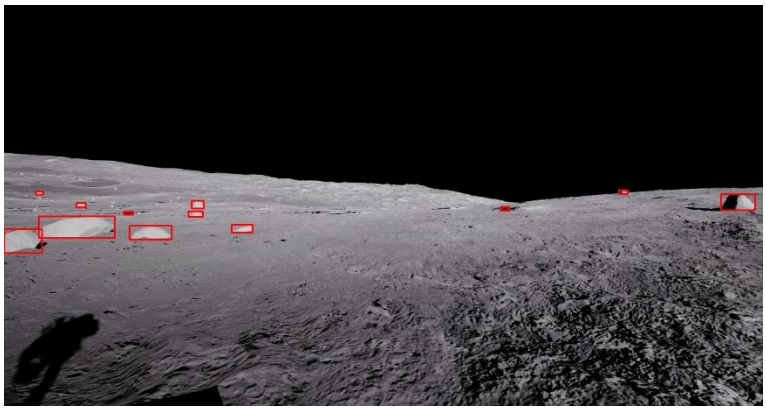

Figure 9: Network inference on a rover image

In addition to hazard detection, loop closure detection and pose graph optimization play a critical role in ensuring long-term consistency in mapping, especially in fixed-zone operations such as planetary construction [35]. To support this functionality, a Bag-of-Words (BoW) visual vocabulary was constructed using ORB descriptors extracted from the same

IMMV[2]-generated dataset. This enables the recognition of previously visited locations based on image similarity, thereby enhancing global map consistency.

The vocabulary was built using the open-source DBoW2 library [37], employing a hierarchical k-means tree with a branching factor of $k = 10$ and a depth of $L = 5$, resulting in up to 10^5 possible visual words.

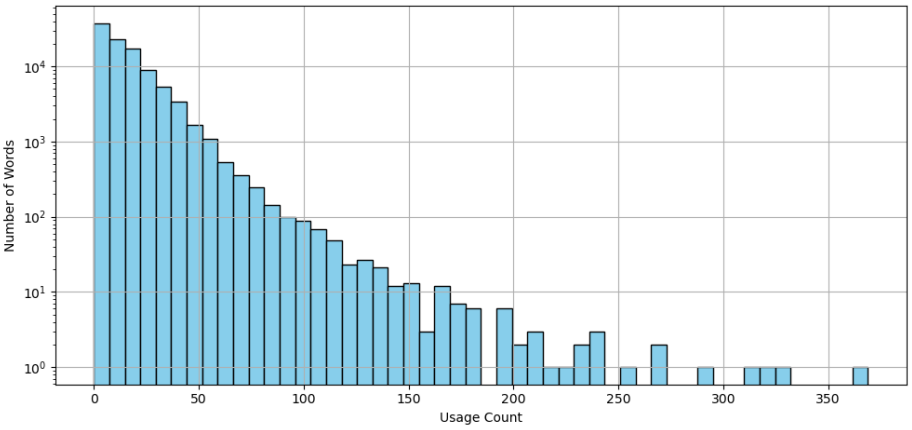

Figure 10: BoW2 vocabulary, word usage count histogram

As shown in Figure 9, the histogram of word usage reveals a relatively balanced frequency distribution, with most visual words appearing fewer than 150 times. Despite the low-texture and unstructured nature of the lunar surface, the vocabulary effectively captured a diverse set of visual features. The resulting long-tailed distribution reflects high feature diversity and limited redundancy, confirming the dataset's suitability for appearance-based loop closure. In practice, the final vocabulary included 10'000 words, with only five unused, indicating strong descriptor space coverage. To further improve discriminability, Term Frequency–Inverse Document Frequency (TF-IDF) weighting and L1-norm scoring were applied. The vocabulary achieved an entropy of 12.959 out of a theoretical maximum of 13.287 (ratio: 0.975), and a sparsity of 0.001 (0.1%), both indicating a well-distributed and informative visual word space.

4. Navigation Filter - Overview

Reliable state estimation in long-range autonomous navigation cannot rely on a single sensing modality. Visual Odometry accumulates scale and orientation errors over time, IMUs suffer from bias drift and wheel odometry is corrupted by slip and uneven terrain. Fusing these complementary sources of information enables each sensor to compensate the failure modes of the others, yielding an estimator that is both locally smooth and globally consistent.

In this section VO, inertial and wheel-encoder data are combined inside a direct feedforward EKF framework. The filter operates on a nine-dimensional state vector to take into account the evolving error bias on the VO

$$\mathbf{x} = [\mathbf{p}^\top \ \mathbf{v}^\top \ \mathbf{b}^\top]^\top \in \mathbb{R}^9$$

where $\mathbf{p} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$ denote position and velocity in the world frame, and $\mathbf{b} \in \mathbb{R}^3$ represents a slowly varying VO bias modelled as a first-order Gauss–Markov process. Wheel odometry provides linear velocity constraints in the body frame, whereas VO contributes pose increments in the camera frame. By expressing both measurement models in a common inertial frame, the EKF recursively updates the state and its covariance, achieving better performance, in terms of accuracy, than IMU+WO localization. The remainder of this section details the probabilistic formulation of the filter, including process and measurement equations, and validation on synthetic datasets.

4.1 Mathematical Implementation

An Extended Kalman Filter fuses Visual Odometry with wheel/IMU velocity to estimate 3-D position, velocity and a slowly varying bias. State vector $\mathbf{x} = [\mathbf{p}\ \mathbf{v}\ \mathbf{b}]^\top \in \mathbb{R}^9$; bias follows a first-order Gauss-Markov law. The filter uses the following observations:

- VO-derived position at 1 Hz: $\mathbf{z}_{\text{VO}} = \mathbf{p}_{\text{VO}}$

- Velocity from Unreal Engine simulation, shown in Figure 11
- Real attitude orientation of the rover considering a disturbing AWGN of covariance $\sigma_a = 0.1$ degrees.

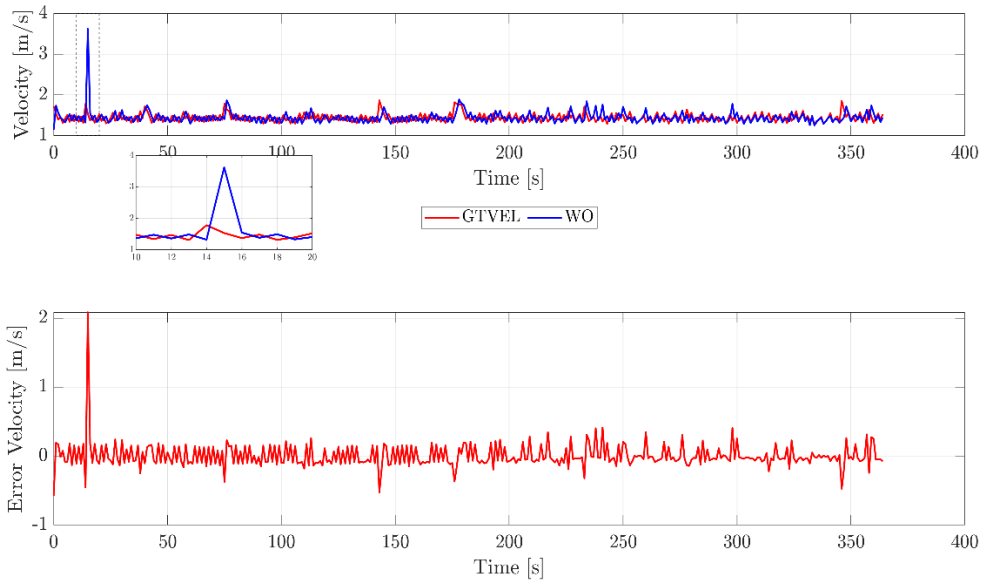

Figure 11: Unreal Engine wheel odometry readings

To let the filter work properly the following parameters are set. Just four process noise parameters on Bias noise and position have been tuned to reach best performance in terms of localization, covariance and robustness, while the following have been set after the following criteria. Firstly $\sigma_v = \sqrt{(\mathbf{E}[(v_{WO} - \mathbf{E}[X])^2]}$ that means basically the covariance of the temporal series of rover encoder readings, secondly from the stereo camera equation in Section 3.2.3 $\sigma^r_{pVO_x} = \frac{Z_{mean}}{f_x}\sigma_{px}$ (and so y) and $\sigma^r_{pVO_z} = \frac{Z^2_{mean}}{f_x B}\sigma_{px}$ (see also [24]). From VO Algorithm settings Z_{mean} is assumed equal to 20 and $\sigma_{px} = 0.5$ after [38]. It is worth noting that this latter assumption is particularly conservative.

Parameters	Symbol	Value
VO time constant	τ_b	80 s
VO Bias noise process noise	σ_{bx}	0.05 m
VO Bias noise process noise	σ_{by}	0.03 m
VO Bias noise process noise	σ_{bz}	0.2 m
VO position process noise	σ_p	0.3 m
Velocity process noise	σ_v	0.2 m/s
VO Mesurement noise std	$\sigma^r_{pVO_x}$	0.02 m
VO Mesurement noise std	$\sigma^r_{pVO_y}$	0.02 m
VO Mesurement noise std	$\sigma^r_{pVO_y}$	0.7 m

To Initialize the filter the state vector $\mathbf{x} \in \mathbb{R}^9$ is set

$$\mathbf{x}_0 = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{v}_0 \\ \mathbf{b}_0 \end{bmatrix}, \quad \mathbf{P}_0 = \text{diag}(0.1\mathbf{I}_3,\ 0.01\mathbf{I}_3,\ 0.5\mathbf{I}_3)$$

The prediction step is composed of a linearized transition model:

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_3 & \Delta t\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \left(1 - \frac{1}{\tau_b}\right)\mathbf{I}_3 \end{bmatrix}$$

And State Noise Compensation process noise Matrix, where Q_u is properly tuned [39].

$$\mathbf{Q_\eta} = \int_{t_0}^{t} \phi\, B Q_u B^T \phi^T d\tau$$

Position is propagated along the estimated body x-axis:

$$\mathbf{v}_{\text{proj}} = v_{\text{noisy}} \cdot \mathbf{R} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{x}_{k|k-1} = \begin{bmatrix} \mathbf{p}_{k-1} + \Delta t \cdot \mathbf{v}_{\text{proj}} \\ \mathbf{v}_{\text{proj}} \\ \left(1 - \frac{1}{\tau_b}\right) \mathbf{b}_{k-1} \end{bmatrix}$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^\top + \mathbf{Q_\eta}$$

The update is carried out using the following measurement model:

$$\mathbf{H} = [\mathbf{I}_3 \quad \mathbf{0}_3 \quad \mathbf{I}_3]$$

Consequently, the update equations:

$$\mathbf{S} = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\top + \mathbf{R}_{\text{VO}}, \quad \mathbf{K} = \mathbf{P}_{k|k-1}\mathbf{H}^\top \mathbf{S}^{-1}$$

$$\mathbf{x}_k = \mathbf{x}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \mathbf{H}\mathbf{x}_{k|k-1}), \quad \mathbf{P}_k = (\mathbf{I}_9 - \mathbf{K}\mathbf{H})\mathbf{P}_{k|k-1}$$

5. Results

In this section quantitative results will be presented for the evaluation of the VBN algorithms.

5.1 First Visual Relative Localization Algorithm

An example of the user visualization window of the first VBN algorithm is presented in Figure 12. In the window a visual representation of the tracked keypoints between two successive frames and the SGM generated disparity map are displayed.

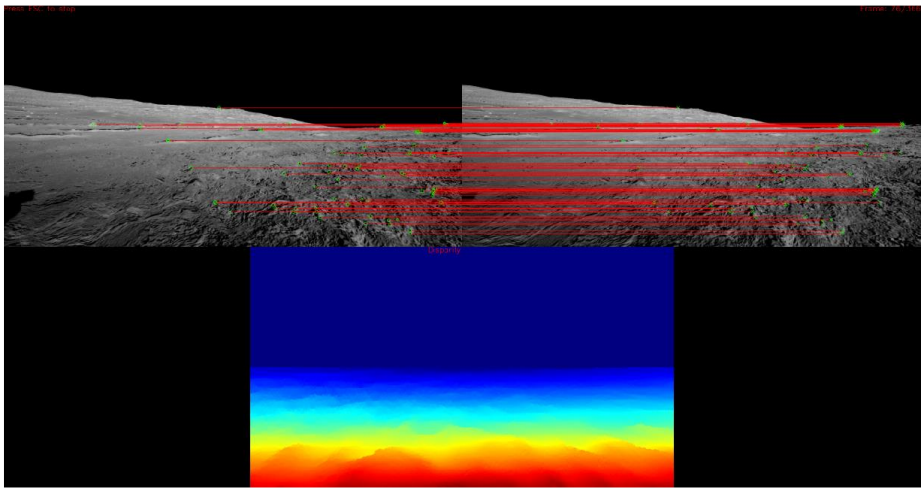

Figure 12: Lunar rover traverse frame

The next figures summarize the EKF behaviour on a representative run.

- **Absolute position error.** Figure 13 shows the 3D Euclidean absolute position error versus time over about 500 m rover traverse, showing 7.6 meters of absolute error (about 1.5% of the distance). It is worth to remark that the beginning of the traverse an unmodeled spiking error due to wheel slippage erroneous reading (see Figure

11 and Figure 13) and how the Sensor Fusion Algorithm manage it. In the same context, it is worth also remarking the spiking VO error, due to an important rover turn, and how the filter manages it (see Figure 13).

- **Consistency check.** In Figure 14 the 3σ envelope derived from the covariance is plotted. It can be noted that the actual error consistently stays inside the 3σ envelope, indicating that the covariance prediction is neither overly optimistic nor overly conservative both in X, Y and Z components.

- **Bias convergence.** Figure 15 depicts the EKF estimate of the Visual Based Localization biases. From the comparison with Figure 16 it is confirmed that the filter is able to understand quite well the behaviour of the error in Visual Based Localization Observables.

- **Trajectory match.** Figure 17 compares the final 3-D EKF trajectory with the ground-truth path. The Fusion clearly outperforms the Inertial integration (WO + IMU).

Overall, the EKF reduces the RMS position error (see comparison with inertial only localization in Figure 13) while maintaining statistical consistency, demonstrating that VO (plus velocity and inertial sensing) is of paramount importance for precise local navigation in a lunar scenario.

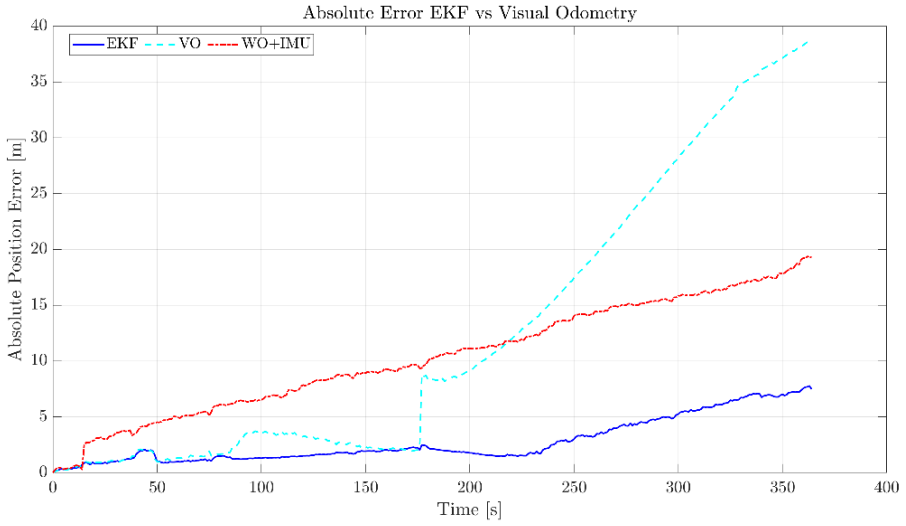

Figure 13: Instantaneous error

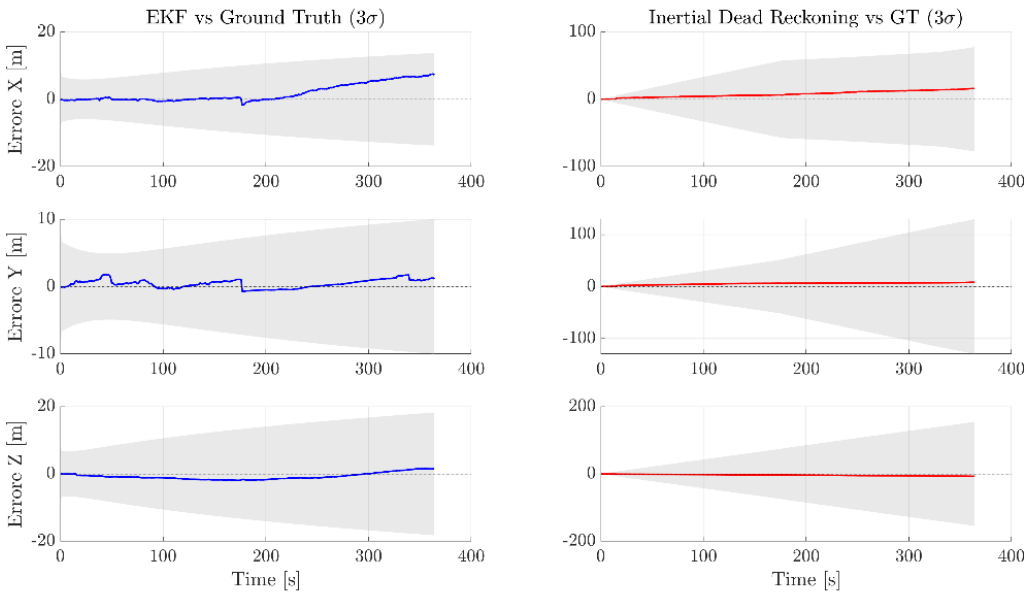

Figure 14: σ_{3D} envelope

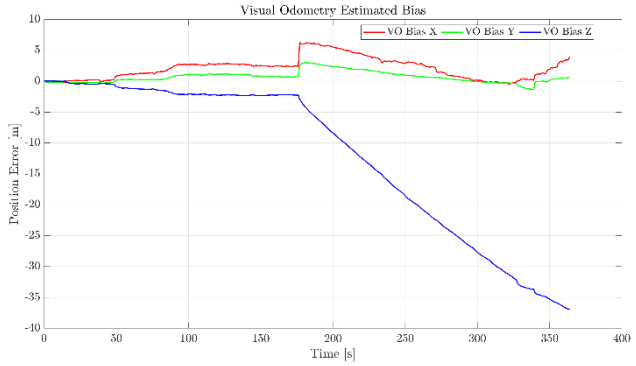

Figure 15: Estimated VO-bias component

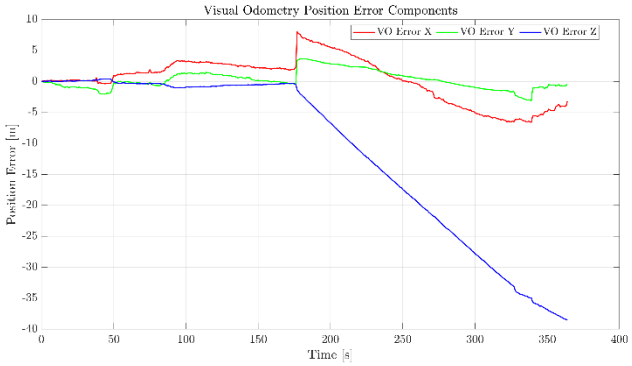

Figure 16: Real VO-bias component

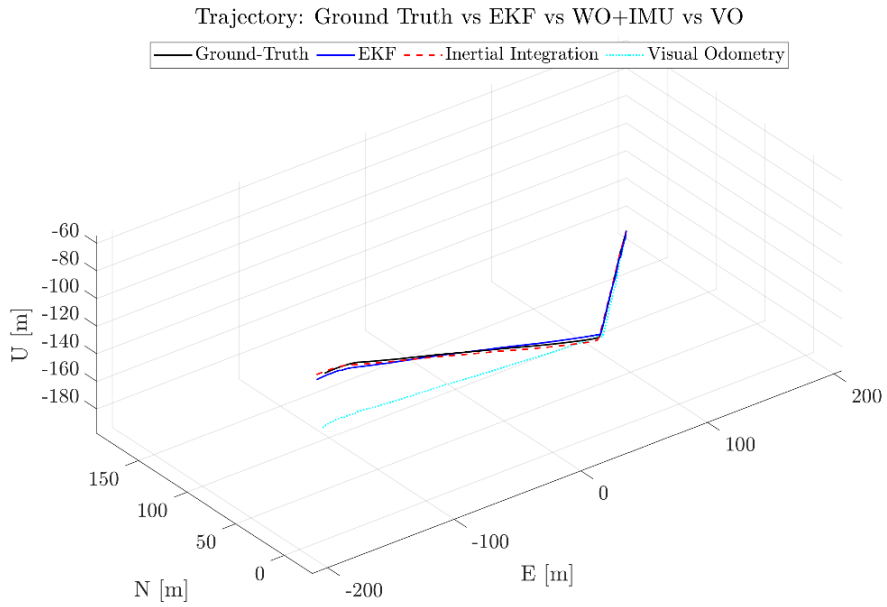

Figure 17: EKF vs ground truth path

5.2 Second Visual Relative Localization Algorithm

As shown in Figure 21, the position error in the X and Y directions remains relatively bounded over time, while the Z-axis exhibits a noticeable drift, reaching nearly 30 meters at the end of the trajectory with a rapid change in the slope when the rover turns left. When fused within the EKF, the algorithm benefits from reduced uncertainty and improved consistency, as illustrated in the 3σ envelope plots in Figure 19. The error remains mostly within the predicted bounds, suggesting good filter convergence. However, compared to the first visual navigation algorithm, the EKF correction is less effective at eliminating bias accumulation—likely due to the more tightly coupled structure of the initial pipeline and its different observable formulation.

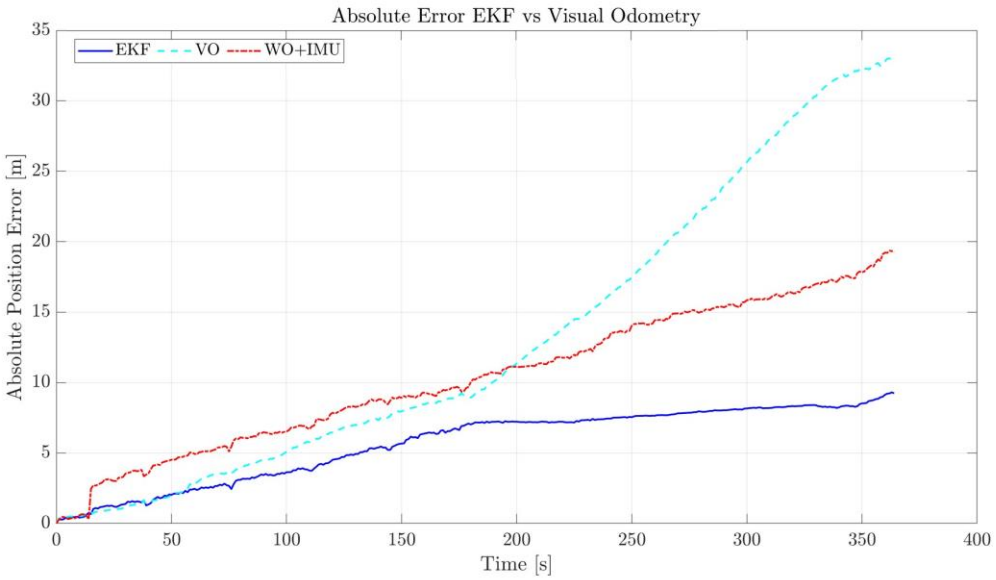

Figure 18: Performance comparison in localization with and without EKF for the second VBN

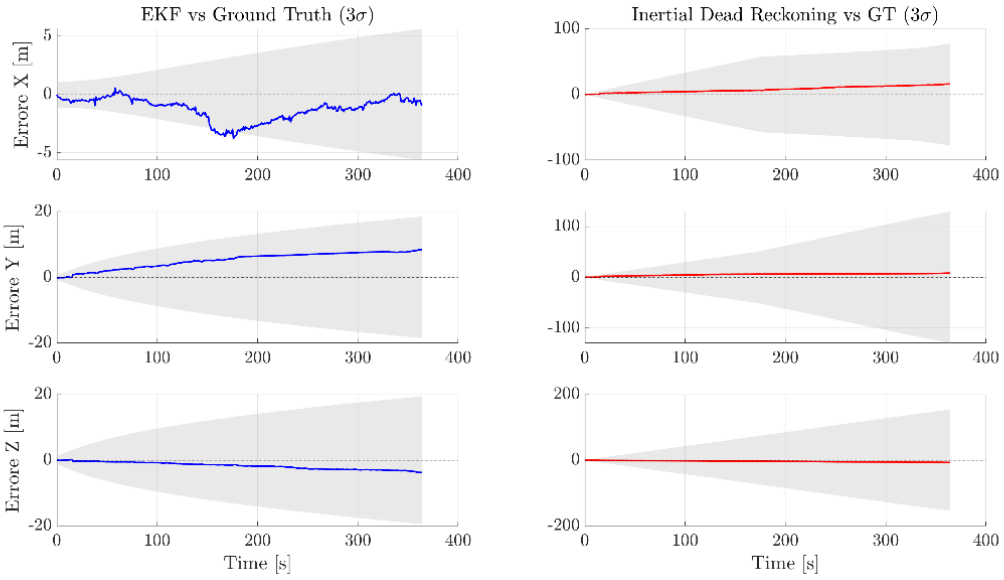

Figure 19: σ_{3D} for the second VBN algorithm

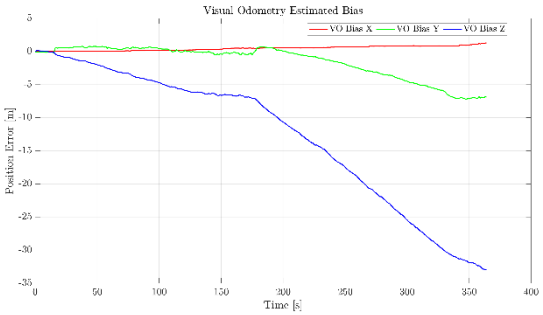

Figure 20: Estimated biases

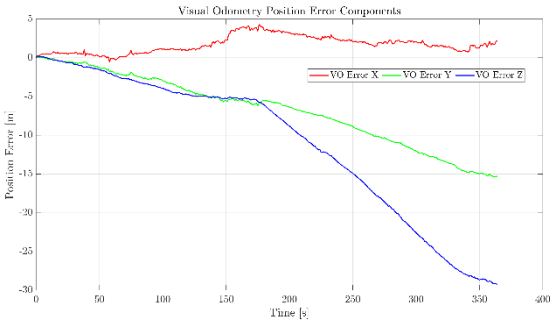

Figure 21: Real biases

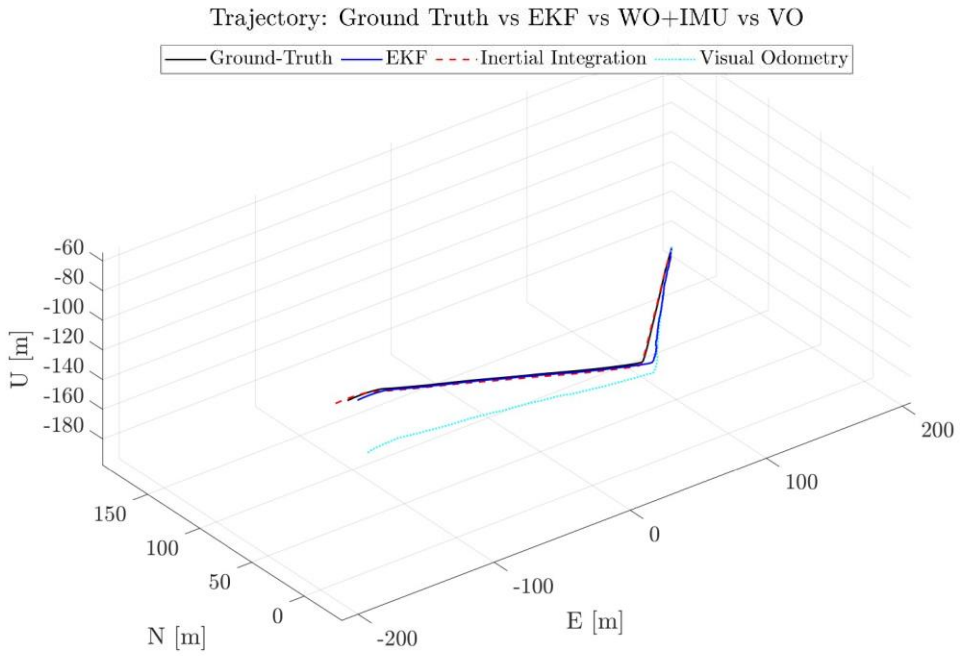

Figure 22: EKF vs ground truth path

6. Conclusions and Future Works

This study presented a multi-sensor fusion approach for rover navigation on the lunar surface, evaluating the performance of two VBN pipelines integrated with synthetic wheel odometry and inertial measurements through EKF. The evaluation was conducted in a high-fidelity digital twin of the lunar south pole environment, generated through the IMMV[2] framework. The digital twin provided centimetre-scale surface geometry and synthetic sensor observables, enabling controlled and repeatable validation of autonomous navigation strategies.

The two tested pipelines demonstrated complementary strengths. The second algorithm achieved improved pure visual odometry performance but exhibited reduced correction efficacy when fused in the EKF, resulting in greater uncompensated drift, particularly in the lateral (Y) direction. Conversely, the first algorithm showed slightly lower VO accuracy but benefitted more substantially from the EKF's state estimation and bias correction, leading to improved global consistency over the entire 500 m reference trajectory. The resulting maximum position error was constrained within 7.5 m, corresponding to 1.5% of the total travelled distance.

These results align well with the expected performance of modern VBN systems in planetary exploration [4]. Although this study does not yet incorporate NAV SATs signals, the current results place the presented VBN pipeline well within the expected performance envelope for near-future lunar navigation architectures.

Future work will focus on extending the presented framework in several directions:

- NAV SATs observables will be integrated in the pipeline to analyze the improvement introduced by a navigation constellation around the moon;
- loop closure detection and global pose graph optimization will be integrated to further suppress accumulated drift, particularly in longer traverses;
- autonomous path planning and decision-making modules will be introduced to enable dynamic interaction with the IMMV[2] simulator through bidirectional control of navigation and hazard avoidance.

These enhancements aim to evolve the current system into a resilient, complete navigation framework capable of operating autonomously in the complex terrain and illumination conditions of the lunar south pole.

References

[1] M. Smith *et al.*, 'The Artemis Program: An Overview of NASA's Activities to Return Humans to the Moon', in *2020 IEEE Aerospace Conference*, Mar. 2020, pp. 1–10. doi: 10.1109/AERO47225.2020.9172323.

[2] European Space Agency (ESA), 'Explore 2040, the european exploration strategy, 2024'. Accessed: May 19, 2025. [Online]. Available: https://esamultimedia.esa.int/docs/HRE/Explore_2040.pdf

[3] L. Bingham, J. Kincaid, B. Weno, N. Davis, E. Paddock, and C. Foreman, 'Digital Lunar Exploration Sites Unreal Simulation Tool (DUST)', in *2023 IEEE Aerospace Conference*, Big Sky, MT, USA: IEEE, Mar. 2023, pp. 1–12. doi: 10.1109/AERO55745.2023.10115607.

[4] Y. Ma, S. Liu, B. Sima, B. Wen, S. Peng, and Y. Jia, 'A precise visual localisation method for the Chinese Chang'e-4 Yutu-2 rover', *Photogramm. Rec.*, vol. 35, no. 169, pp. 10–39, Mar. 2020, doi: 10.1111/phor.12309.

[5] R. Ghosh, S. Tomar, C. S. Mhatre, K. Sumithra, B. K. G.V.P, and M. S. Siva, 'Path Planning for the Pragyan Rover: Experiences and Challenges', in *2024 International Conference on Space Robotics (iSpaRo)*, Jun. 2024, pp. 70–75. doi: 10.1109/iSpaRo60631.2024.10687930.

[6] G. Cifani, A. Darrau, F. Rometsch, G. Alvarez, L. Duvet, and R. Biesbroek, 'Argonaut: ESA's Versatile Lunar Lander Enabling Multiple Moon Missions', *IAF Hum. Spacefl. Symp.*, doi: 10.52202/078364-0005.

[7] 'Lunar Polar Exploration(LUPEX)', JAXA Human Spaceflight Technology Directorate. Accessed: May 19, 2025. [Online]. Available: https://humans-in-space.jaxa.jp/en/biz-lab/tech/lupex/

[8] S. Giannattasio, L. Andolfi, M. Brancati, A. M. Di Donna, G. Tomasicchio, and L. Ostrogovich, 'An advanced tool for interactive mission modeling & visualization/validation of space-based scenarios', in *IAC-24,B5,1,x85403*,

[9] G. Tomasicchio *et al.*, 'Multi-Sensor Fusion and Resilient PVT Techniques for Safe Lunar Landing Missions', presented at the Proceedings of the 37th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2024), 264 2024, pp. 3125–3143. doi: 10.33012/2024.19849.

[10] L. Ostrogovich, R. Del Prete, G. Tomasicchio, N. Longépé, and A. Renga, 'A Dual-Mode Approach for Vision-Based Navigation in a Lunar Landing Scenario', in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2024, pp. 6799–6808. doi: 10.1109/CVPRW63382.2024.00673.

[11] G. A. Neumann, 'Lunar Orbiter Laser Altimeter Raw Data Set'. NASA Planetary Data System, 2010. doi: 10.17189/1520642.

[12] E. H. Blumenfeld *et al.*, '3D Virtual Astromaterials Samples Collection of NASA's Apollo Lunar and Antarctic Meteorite Samples to be an Online Database to Serve Researchers and the Public', presented at the 50th Lunar and Planetary Science Conference (2019), Accessed: May 19, 2025. [Online]. Available: https://www.hou.usra.edu/meetings/lpsc2019/pdf/3056.pdf

[13] 'Virtual Shadow Maps in Unreal Engine | Unreal Engine 5.1 Documentation | Epic Developer Community', Epic Games Developer. Accessed: May 19, 2025. [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/lighting-the-environment

Company General Use

[14] ‘Lumen Global Illumination and Reflections in Unreal Engine | Unreal Engine 5.1 Documentation | Epic Developer Community’, Epic Games Developer. Accessed: May 19, 2025. [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/global-illumination

[15] ‘Nanite Virtualized Geometry in Unreal Engine | Unreal Engine 5.1 Documentation | Epic Developer Community’. Accessed: May 19, 2025. [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/nanite-virtualized-geometry-in-unreal-engine?application_version=5.1

[16] ‘Vehicles in Unreal Engine | Unreal Engine 5.5 Documentation | Epic Developer Community’, Epic Games Developer. Accessed: May 19, 2025. [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/vehicles-in-unreal-engine

[17] ‘Perseverance Rover Components - NASA Science’. Accessed: May 19, 2025. [Online]. Available: https://science.nasa.gov/mission/mars-2020-perseverance/rover-components/

[18] ‘Mars Perseverance Rover, 3D Model - NASA Science’. Accessed: May 19, 2025. [Online]. Available: https://science.nasa.gov/resource/mars-perseverance-rover-3d-model/

[19] J. N. Maki *et al.*, ‘The Mars 2020 Engineering Cameras and Microphone on the Perseverance Rover: A Next-Generation Imaging System for Mars Exploration’, *Space Sci. Rev.*, vol. 216, no. 8, p. 137, Nov. 2020, doi: 10.1007/s11214-020-00765-9.

[20] iMAR Navigation, ‘iNAV-RQH: Inertial Gyro Navigation System (ring laser gyro based), 0.001 deg/sqrt(hr)’. Accessed: May 20, 2025. [Online]. Available: https://www.imar-navigation.de/de/produkte-uebersicht/product-overview-by-product/item/inav-rqh-inertial-laser-gyro-navigation-system

[21] W. D. Carrier III, G. R. Olhoeft, and W. Mendell, ‘Physical Properties of the Lunar Surface’, in *Lunar Sourcebook, A User’s Guide to the Moon*, 1991, pp. 475–594. Accessed: May 19, 2025. [Online]. Available: https://ui.adsabs.harvard.edu/abs/1991lsug.book..475C

[22] J.-Y. Bouguet, ‘Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm’, *Intel Corp.*, vol. 5, no. 1–10, p. 4, 2001.

[23] H. Hirschmuller, ‘Stereo Processing by Semiglobal Matching and Mutual Information’, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008, doi: 10.1109/TPAMI.2007.1166.

[24] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[25] V. Lepetit, F. Moreno-Noguer, and P. Fua, ‘EPnP: An Accurate O(n) Solution to the PnP Problem’, *Int. J. Comput. Vis.*, vol. 81, no. 2, pp. 155–166, Feb. 2009, doi: 10.1007/s11263-008-0152-6.

[26] M. A. Fischler and R. C. Bolles, ‘Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography’, *Commun ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, doi: 10.1145/358669.358692.

[27] ‘Computer Vision Toolbox’. Accessed: May 19, 2025. [Online]. Available: https://it.mathworks.com/products/computer-vision.html

[28] R. Mur-Artal and J. D. Tardós, ‘ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras’, *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[29] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, ‘ORB: An efficient alternative to SIFT or SURF’, in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.

[30] M. Muja and D. G. Lowe, ‘Fast Matching of Binary Features’, in *2012 Ninth Conference on Computer and Robot Vision*, May 2012, pp. 404–410. doi: 10.1109/CRV.2012.60.

[31] ‘estworldpose - Estimate camera pose from 3-D to 2-D point correspondences - MATLAB’. Accessed: May 19, 2025. [Online]. Available: https://it.mathworks.com/help/vision/ref/estworldpose.html

[32] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, ‘Complete solution classification for the perspective-three-point problem’, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 930–943, Aug. 2003, doi: 10.1109/TPAMI.2003.1217599.

Company General Use

[33] P. H. S. Torr and A. Zisserman, ‘MLESAC: A New Robust Estimator with Application to Estimating Image Geometry’, *Comput. Vis. Image Underst.*, vol. 78, no. 1, pp. 138–156, Apr. 2000, doi: 10.1006/cviu.1999.0832.

[34] ‘worldpointset - Manage 3-D to 2-D point correspondences - MATLAB’. Accessed: May 19, 2025. [Online]. Available: https://it.mathworks.com/help/vision/ref/worldpointset.html

[35] ‘imageviewset - Manage data for structure-from-motion, visual odometry, and visual SLAM - MATLAB’. Accessed: May 19, 2025. [Online]. Available: https://it.mathworks.com/help/vision/ref/imageviewset.html

[36] G. Jocher, J. Qiu, and A. Chaurasia, *Ultralytics YOLO*. (Jan. 2023). Python. Accessed: May 19, 2025. [Online]. Available: https://github.com/ultralytics/ultralytics

[37] D. Galvez-López and J. D. Tardos, ‘Bags of Binary Words for Fast Place Recognition in Image Sequences’, *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012, doi: 10.1109/TRO.2012.2197158.

[38] ‘Evaluating the Accuracy of Single Camera Calibration - MATLAB & Simulink’. Accessed: May 20, 2025. [Online]. Available: https://it.mathworks.com/help/vision/ug/evaluating-the-accuracy-of-single-camera-calibration.html

[39] B. Schutz, B. Tapley, and G. H. Born, *Statistical orbit determination*. Elsevier, 2004.

Notes

During the preparation of this work the author(s) used ChatGPT4 in order to improve text readability and language. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Company General Use