doi: 10.82124/CEAS-GNC-2026-XYZ CEAS-GNC-2026-XYZ

Enhanced Sensor Fusion for Autonomous Navigation in GPS-Denied Unknown Environments of Space Robots

Niccolò Goretti MSc student in Space Engineering, Department of Aerospace Science and Technology (DAER), Politecnico di Milano, Milano, Italy.
niccolo.goretti@mail.polimi.it

Annachiara Ippolito PhD student in Aerospace Engineering, Department of Aerospace Science and Technology (DAER), Politecnico di Milano, Milano, Italy.
annachiara.ippolito@polimi.it

Mauro Massari Associate Professor of Space Systems, Department of Aerospace Science and Technology (DAER), Politecnico di Milano, Milano, Italy.
mauro.massari@polimi.it

ABSTRACT

Exploring unknown, GPS-denied environments poses critical challenges, especially in space missions where autonomous navigation is essential. Simultaneous Localization and Mapping (SLAM) enables Unmanned Ground Vehicles (UGVs) to navigate autonomously in such conditions, but reliable localization and mapping must overcome sensor noise, environmental uncertainty, and drift. This work presents a GPS-free, multi-sensor SLAM system for space UGVs that integrates data from a Light Detection and Ranging (LiDAR) sensor, a Monocular Camera (MC), and an Inertial Measurement Unit (IMU) through enhanced sensor fusion. A modified Graph-Based (GB) method fuses MC and IMU localization estimates to reduce drift, while an Extended Kalman Filter (EKF) processes LiDAR-based localization in a second fusion stage, ensuring accurate mapping. Simulations conducted in a virtual environment validate the proposed method under realistic conditions. The results show significant reductions in positional drift, improved localization robustness, and superior performance compared to existing SLAM techniques. By combining complementary sensors in a modular framework, the approach improves SLAM for autonomous navigation in unstructured terrains and demonstrates strong potential for planetary exploration and other GPS-denied missions.

Keywords: Autonomous Navigation, GPS-denied Environments, Space Rovers, Unmanned Ground Vehicles, Sensor Fusion, Simultaneous Localization and Mapping

Nomenclature

δ_{rot1}	=	Initial rotation command (odometry model)
δ_{trans}	=	Translation command (odometry model)
δ_{rot2}	=	Final rotation command (odometry model)
δ_{rot1_n}	=	Noisy initial rotation command
δ_{trans_n}	=	Noisy translation command

δ_{rot2_n}	=	Noisy final rotation command
$\epsilon_{\text{rot1}}, \epsilon_{\text{rot2}}, \epsilon_{\text{trans}}$	=	Gaussian noise terms for control commands
x, y	=	Position in Absolute Reference Frame
θ	=	Heading angle (yaw about vertical axis)
s	=	Along-track displacement in Relative Reference Frame
n	=	Cross-track displacement in Relative Reference Frame
h	=	Vertical axis in Relative Reference Frame
ψ, θ, ϕ	=	Euler angles (yaw, pitch, roll)
$\boldsymbol{p}$	=	State / pose vector (x, y, θ)
$\hat{\boldsymbol{p}}$	=	Optimized pose (Graph-Based fusion)
$\boldsymbol{p}_{\text{IMU}}$	=	Pose estimate from Inertial Measurement Unit
$\boldsymbol{p}_{\text{MC}}$	=	Pose estimate from Monocular Camera
$\mathbf{W}_{\text{IMU}}$	=	Confidence weight matrix for Inertial Measurement Unit
$\mathbf{W}_{\text{MC}}$	=	Confidence weight matrix for Monocular Camera
P	=	State covariance matrix
Q	=	Process noise covariance matrix
R	=	Measurement noise covariance matrix
$\boldsymbol{p}_{\text{pred}}$	=	Predicted state (Extended Kalman Filter)
P_{pred}	=	Predicted covariance (Extended Kalman Filter)
$\boldsymbol{z}$	=	LiDAR measurement vector
H	=	Observation model matrix
K	=	Kalman gain
$\boldsymbol{p}_{\text{est}}$	=	Estimated state after Extended Kalman Filter update
$\overline{\text{err}}_s$	=	Mean along-track error
$\overline{\text{err}}_n$	=	Mean cross-track error
$\overline{\text{err}}_\theta$	=	Mean heading error

1 Introduction

Space exploration continues to drive scientific and technological advances, with the Moon serving as a key target and testbed for deep-space missions. Lunar subsurface structures such as lava tubes and impact crater caves offer natural shelters from radiation, micrometeoroids, and thermal extremes, while preserving valuable geological records [1–5]. However, autonomous navigation in these environments faces significant challenges due to the lack of global references (GPS, magnetic fields, lighting), unknown and unmapped terrain, irregular features, dust, temperature gradients, communication delays, and limited computational resources [6, 7]. These factors underscore the need for efficient real-time localization and mapping solutions.

To address these challenges, this work proposes a resource-efficient multi-sensor Simultaneous Localization and Mapping (SLAM) framework for GPS-denied, unstructured environments, motivated by lunar cave exploration but applicable to broader scenarios. The approach fuses data from an Inertial Measurement Unit (IMU), a Light Detection and Ranging (LiDAR), and a Monocular Camera (MC) to exploit complementary strengths. A hybrid wheeled-legged Unmanned Ground Vehicle (UGV) is chosen to enhance terrain adaptability, with a novel sensor fusion architecture combining a modified graph-based (GB) method and an Extended Kalman Filter (EKF) to improve robustness and accuracy. Validation is performed in a high-fidelity simulation pipeline using Blender™, Unreal Engine™, MATLAB™, and Simulink™. The study focuses on planar motion to reduce complexity while reflecting key lunar subsurface navigation challenges.

SLAM, a core tool for GPS-denied navigation, has evolved from probabilistic filters to graph-based and hybrid methods [8–10], while the need for consistency in pose and landmark estimation has been emphasized since early work [11]. Despite advances, many systems rely on single sensors and struggle in unstructured, feature-scarce environments like lunar caves [12, 13]. This motivates the proposed multi-sensor fusion framework, designed to overcome such limitations by integrating diverse sensing modalities and accommodating space-specific constraints.

Hardware and sensor choices critically impact navigation performance. Common alternatives, such as dead reckoning, beacon-based, time-of-flight (ToF), landmark, and vision-based methods, each have drawbacks in subsurface scenarios [14]. Multi-sensor fusion combining visual, inertial, and ToF-based sensing has shown superior results [15, 16]. Accordingly, this work adopts an IMU, LiDAR, and monocular camera with artificial lighting to enable robust perception.

Mobility is addressed by selecting a hybrid wheeled-legged UGV, balancing terrain adaptability and efficiency [17]. While UAV-UGV cooperative systems are effective in terrestrial GPS-denied exploration, lunar aerial mobility demands alternative propulsion such as cold gas thrusters due to the lack of atmosphere, informing the proposed multi-platform architecture.

This paper is structured to progressively build upon the proposed approach, guiding the reader through the proposed methodology, experimental results, and concluding discussions. Section 2 details the proposed methodology, including the system modular configuration and the two-layer sensor fusion framework. Section 3 outlines the simulation workflow and presents the corresponding results after each processing stage. It then compares the proposed method against a standard EKF based on odometry and concludes with a sensitivity analysis on the EKF noise parameters. Section 4 concludes the paper with a summary of findings and future research directions.

2 Methods

The proposed methodology addresses the key challenges of autonomous navigation in GPS-denied environments by adopting a modular and scalable multi-sensor SLAM framework. Designed to ensure real-time operation, robustness, and adaptability, the system integrates data from an IMU, a LiDAR, and a MC within a loosely coupled architecture. Validation is performed through a high-fidelity simulation pipeline that combines photorealistic rendering, physics-based sensor emulation, and algorithmic processing. A concise summary of the hardware configuration and the modular architecture developed for data integration and processing is presented here. With this foundation established, the focus of this section shifts to a detailed description of the navigation system design and operational flow.

2.1 Reference Scenario Definition

The reference scenario builds upon the key findings discussed earlier in the introduction, serving as a foundation for introducing the modular navigation system underlying the proposed SLAM framework.

2.1.1 Hardware Setup

The proposed system is composed of a deployer, a UAV, and a UGV, each contributing to localization, communication, exploration, and environmental monitoring. The deployer functions as a reference point, thus as a landmark, and as a communication hub, equipped with dual radio transceivers that act as beacons and enable coordination across platforms. It also supports logistical functions such as sample storage, power supply, and environmental monitoring.

The UAV enhances mapping through high-resolution imaging and atmospheric sensing and may also assist localization by deploying beacons. The UGV, equipped with a hybrid wheeled-legged configuration, autonomously navigates using SLAM, performs terrain monitoring, and collects soil and rock samples.

Each component is outfitted with a specific sensor suite: the deployer includes dual radio transceivers, the UAV carries a Stereo Camera (SC), an IMU, and a transceiver, and the UGV is equipped with MC, IMU, LiDAR, and a radio transceiver. This sensor distribution supports robust, cooperative aerial-ground exploration in GPS-denied environments.

2.1.2 Navigation System Architecture

The architecture enabling real-time localization, mapping, and navigation is illustrated in Fig. 1. It integrates data from all platforms to ensure continuity and adaptability and is composed of six modules:

- **Data Interface Module (DIM):** Handles the acquisition and preprocessing of raw sensor data, ensuring storage and compatibility.
- **Sensor Fusion Module (SFM):** Integrates multiple sensor inputs onboard the UGV to estimate its local and global pose while generating a preliminary environmental map.
- **Localization Module (LM):** Uses the UGV's local pose from SFM and refines it through triangulation with radio beacons, improving accuracy.
- **Mapping Module (MM):** Enhances the initial map coming from the SFM by incorporating aerial data from the UAV's SC.
- **Path Planning Module (PPM):** Determines optimal navigation routes based on localization and mapping data, dynamically adjusting paths in response to environmental changes.
- **Control Module (CM):** Converts planned trajectories from PPM into precise actuator commands, guiding the UGV along its path while compensating for control noise.

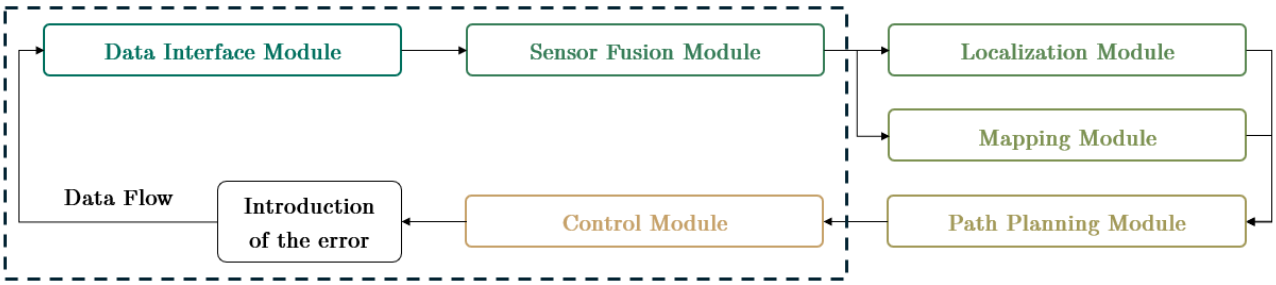


Fig. 1 Modular architecture designed for the navigation system: data flow and module interactions (developed modules in the dashed box)

A key design concern is control noise in CM, due to actuator imprecision. To reduce reliance on noisy odometry, the system emphasizes advanced fusion techniques using environmental data. This structure ensures modularity, fault-tolerance, and robustness in GPS-denied environments.

As anticipated, the core focus of this paper is the development of an effective SLAM method tailored for a UGV. With reference to Fig. 1, the implementation concentrates on the modules supporting the early stages of navigation, specifically the DIM, the SFM, and the CM, including the modeling of control error. Therefore, the following paragraphs focus on the detailed design of these key modules.

Concerning the PPM, which dynamically determines the most efficient navigation routes, it is used in this work solely to provide control inputs as constraints for the SLAM process and is not developed in detail. Similarly, the LM, which integrates the UGV's local pose from the SFM with triangulation data from radio beacons to enhance localization precision, and the MM, which refines the preliminary environmental map from the SFM with aerial data from the UAV's SC, are treated as downstream components. While integral to the system, the detailed implementation of LM and MM is beyond the scope of this paper.

2.2 Design Breakdown

With the navigation architecture and sensor integration strategy well-defined, the focus now shifts to the structured workflow adopted for implementing the navigation algorithm. This section begins by describing the key modules designed for this work, starting from the control commands that are provided to the UGV. These commands govern its movement, enabling the initiation of sensor data acquisition. Subsequently, the focus shifts to the fusion of sensor data with the aim of achieving optimal localization accuracy. Finally, a brief explanation of the fundamental reference frames used in the simulation is introduced, serving as preparation for the detailed simulation description and ensuring clarity across the various stages of the workflow.

2.2.1 CM - Control Commands and Error

The CM serves as the actuator interface of the system, issuing movement commands to the UGV based on navigation inputs. In this study, no external inputs are provided to the CM. Instead, both the trajectory and the associated control commands are derived from a predefined path simulating the expected output of the PPM. These signals are manually defined to reflect the nominal system behavior and are expressed as odometry-based motion instructions, including initial rotation (δ_{rot1}), translation (δ_{trans}), and final rotation (δ_{rot2}), computed from the desired trajectory using a standard odometry motion model [8].

To emulate real-world conditions, where mechanical imperfections and sensor inaccuracies introduce noise into control signals, an artificial error is added to the control commands. This is achieved by superimposing Gaussian noise with a predefined amplitude on the ideal control inputs. Specifically, the translation and rotational components ($\delta_{\text{trans}}, \delta_{\text{rot1}}, \delta_{\text{rot2}}$) are perturbed to simulate deviations from the nominal trajectory. The addition of this noise produces a second set of control inputs representing a noisy scenario, alongside the original ideal commands. Formally, the noisy control commands are generated by adding random perturbations drawn from a normal distribution to each component of the control inputs:

$$\begin{gathered} \delta_{\text{trans}_\text{n}} = \delta_{\text{trans}} + \epsilon_{\text{trans}}, \quad \delta_{\text{rot1}_\text{n}} = \delta_{\text{rot1}} + \epsilon_{\text{rot1}}, \\ \delta_{\text{rot2}_\text{n}} = \delta_{\text{rot2}} + \epsilon_{\text{rot2}} \end{gathered} \tag{1}$$

where each ϵ term is drawn from a zero-mean Gaussian distribution with a standard deviation scaled by a predefined amplitude. This value corresponds to a fixed standard deviation of 0.01 m for translations and 0.01 rad for rotations, which amounts to 5–10% of the average command magnitude in typical trajectories. This process generates a reference trajectory that deviates from the ideal path, simulating the impact of actuator inaccuracies.

Consequently, the CM outputs two sets of control commands: the ideal set, representing perfect actuation along the planned path, and the noisy set, representing a more realistic scenario with perturbations. These control inputs are then used to drive the UGV in the simulation, ensuring both nominal and perturbed trajectories are available for subsequent processing by the navigation filter.

2.2.2 DIM - Raw Data Collection

The DIM acts as the data acquisition gateway of the system, collecting raw measurements from the UGV's onboard sensors. It operates in parallel with the CM, receiving both ideal and perturbed control inputs to initiate simulation. The ideal control inputs serve to initialize the UGV's movement in the simulation, allowing the sensors to start capturing data along the predefined trajectory. In contrast, the noisy controls are used as a baseline for traditional EKF processing, providing a reference against which the performance of the proposed SLAM approach can later be assessed.

During the simulation, the DIM captures and organizes data from the sensor suite, including LiDAR point clouds, MC images, and IMU data. These raw datasets serve as inputs for subsequent localization and mapping. Specifically, the LiDAR provides dense 3D spatial information about the environment, the MC delivers high-resolution visual frames, and the IMU supplies measurements of linear acceleration and angular velocity that reflect the rover's dynamics. The data acquisition process implemented in the DIM ensures that each sensor's data stream is properly captured and stored, ready for use in the SFM.

Accordingly, the DIM outputs the raw sensor datasets required for downstream processing: LiDAR point clouds, MC images, and IMU data.

2.2.3 SFM - Navigation Filter Outline

The navigation filter is implemented in the SFM using a multi-layer loose coupling approach, illustrated in Fig. 2. This design enables each sensor to operate independently while producing individual pose estimates that are selectively fused. Compared to tightly coupled methods, this structure simplifies integration, troubleshooting, and modular adaptation, enabling the replacement or modification of sensor modules without extensive system adjustments [18].

In the first filtering layer, independent pose estimates are generated from the three primary sensors. At this stage, each one operates autonomously, ensuring modularity while establishing a solid foundation for data fusion.

The second filtering layer refines sensor data through a two-loop fusion process. In the first loop, the IMU and MC pose estimates are fused using a modified GB method that employs a weighted approach to optimize their respective contributions. In the second loop, an EKF refines the unified estimate obtained from the first loop by integrating the independent LiDAR pose estimate, producing the final fused trajectory with improved accuracy.

Overall, the SFM receives as input the raw data streams collected by the DIM, comprising LiDAR point clouds, MC images, and IMU measurements. These data enable the computation of independent pose estimates for each sensor, which are subsequently fused through the described process. The module outputs the fused pose estimate and a preliminary map of the environment generated from LiDAR data. These outputs provide essential inputs for the downstream LM and MM, which use them as a baseline.

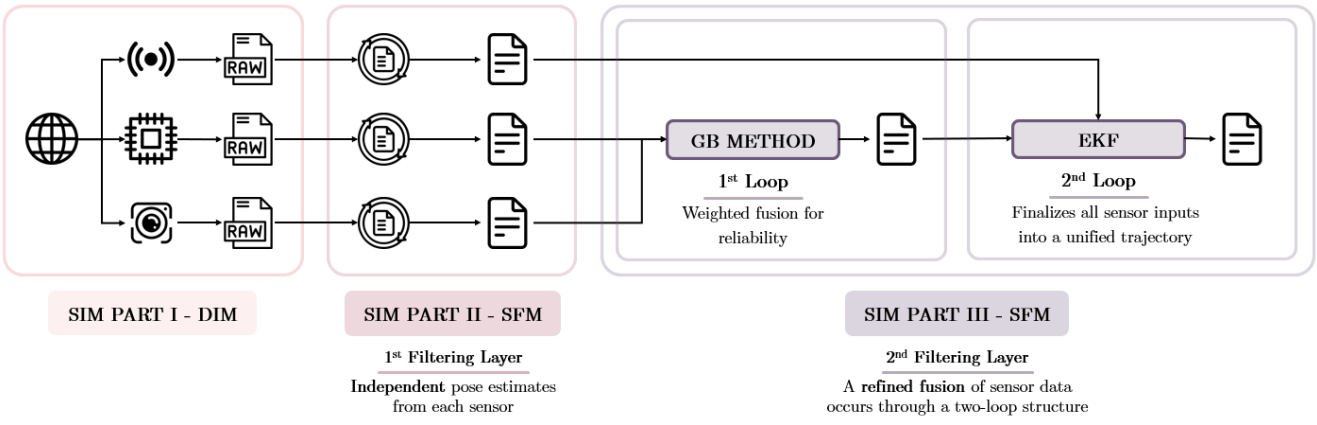


Fig. 2 Navigation filter outline: multi-layer loose coupling

2.2.4 Reference Frames

The simulation framework uses four reference frames, each with a specific role:

- **Absolute Reference Frame (ARF)**: This frame is defined in the context of the Automated Driving Toolbox [19]. Its conventions follow the official documentation, which describes the Earth-fixed

reference frame (x, y, z), as inertial and adhering to the right-hand rule. The axes are defined such that x points in the vehicle's initial forward direction, parallel to the ground plane; y is orthogonal to x, also parallel to the ground plane; and z points upward, perpendicular to the ground plane.

- **World Coordinate System (WCS)**: Unreal Engine™ uses a world coordinate system (X, Y, Z), that is also inertial but follows the left-hand rule. Its origin is located at the center of the simulation environment. Although its axes are oriented similarly to the ARF, the difference in handedness necessitates a transformation. To ensure consistency when integrating data from MATLAB™/Simulink™ into Unreal Engine™, it is necessary to invert the y-axis and all rotations around it.
- **Relative Reference Frame (RRF)**: The vehicle-fixed coordinate system, denoted as (s, n, h), is non-inertial and moves with the rover. The axes are defined as s (Along-Track), representing the forward direction; n (Cross-Track), the leftward direction, perpendicular to s; and h (Height), the upward direction. The associated rotational degrees of freedom are ψ (Yaw), which is the rotation about the h-axis; θ (Pitch), the rotation about the n-axis; and ϕ (Roll), the rotation about the s-axis.
- **Pattern Reference Frame (PRF)**: A camera-relative reference frame (X_P, Y_P, Z_P) is used for MC data. The nomenclature used mirrors the fact that, to estimate the parameters of a MC, calibration is typically performed using multiple images of a calibration pattern, such as a checkerboard. In this frame, X_P points to the right in the image plane; Y_P points downward in the image plane; and Z_P points outward from the image plane, meaning towards the camera. Since the MC is onboard the rover, its reference frame is inherently linked to the vehicle's motion.

As anticipated in the introduction, the study focuses on planar motion to simplify the problem and highlight key challenges. Consequently, axes perpendicular to the simulation plane in inertial reference frames are disregarded. This assumption aligns with the concept of a 2D pose, a widely used term in robotics and SLAM to describe a vehicle's state in terms of position and orientation. A 2D pose characterizes the vehicle's position on a plane along with its heading angle. It is expressed in two different reference frames:

- **Absolute Pose**: Defined in the ARF as (x, y, θ), where x, y represent the vehicle's absolute position, and θ represents the heading angle, defined as the yaw rotation about the h-axis of the RRF.
- **Body-Centric Pose**: Defined in the RRF as (s, n, θ), where s is the forward displacement (Along-Track), n the lateral displacement (Cross-Track), and θ the yaw angle about the vertical axis.

The absolute pose (x, y, θ) is primarily used for trajectory estimation and visualization, assuming an initial condition of $(x, y) = (0, 0)$ and $\theta = 0$. This establishes a global reference for position and heading as the rover navigates through the environment. However, to assess deviations from the intended trajectory, a transformation to the body-centric pose (s, n, θ) is employed. This representation facilitates the evaluation of the rover's motion relative to its own frame, offering a clearer understanding of its alignment with the planned path. By combining absolute and body-centric poses, the navigation system effectively reconstructs the rover's trajectory while ensuring an intuitive interpretation of its movement within the simulation. This dual representation enhances localization accuracy and provides a structured framework for analyzing positioning errors.

3 Results and Discussion

In this section, the simulation setup is detailed and the results achieved are evaluated. The discussion begins with an overview of the simulation tools and their specific roles. The analysis then follows the structure outlined in Fig. 2, illustrating how each module is implemented and examining the corresponding outputs in the order they were introduced. A comparative analysis with a standard odometry-based EKF is then provided, followed by an exploration of the EKF's sensitivity to variations in the Q and R matrices.

3.1 Simulation Framework

The simulation framework includes Blender™, Unreal Engine™, MATLAB™, and Simulink™.

Blender™ is used to refine the rover's 3D model and perform camera calibration using a checkerboard pattern. The selected MC is the FLIR™ Blackfly S[1], featuring a 90° horizontal field of view (hFOV) and 1.3 MPx resolution. Calibration is carried out in MATLAB™ using synthetic images from Blender™ to estimate intrinsic and extrinsic parameters.

Unreal Engine™ provides a high-fidelity simulation environment and handles real-time interaction between the rover and the virtual world. The environment is kept simple and flat to allow controlled testing. The imported rover model is configured with blueprints for the MC and LiDAR actors, which interact with Simulink™ via predefined tags. The IMU, on the other hand, is not simulated within Unreal Engine™. Instead, it is modeled in Simulink™ to enable a dedicated acquisition rate, independent of the rendering frequency, thus avoiding the inaccuracies associated with frame-based sampling.

Simulink™ handles rover dynamics and sensor data acquisition, acting as the interface between Unreal Engine™ and MATLAB™ via the Vehicle Dynamics Blockset [20] and the Automated Driving Toolbox. A global simulation model organizes all components, with specific blocks for sensor transforms and scene setup. Sensor positions are defined relative to the rover, with LiDAR placed above the MC to maximize field of view (FOV) and reduce interference. The final vertical field of view (vFOV) is limited to 30° through an outlier removal filter.

MATLAB™ acts as the computational core, managing SLAM execution, trajectory generation, and data processing. Its integration with the other platforms ensures coherence across the full pipeline.

3.2 Simulation Part I: Data Harvesting

Data generation is divided into simulation setup and sensor data collection. The first phase defines waypoints on a 2D scene, smooths the trajectory, and generates velocity and heading inputs using the odometry motion model, previously detailed in Section 2.2.1, which decomposes movement into initial rotation, translation, and final rotation. To ensure smooth heading transitions, yaw angles in the [0°, 360°] range are unwrapped in order to avoid discontinuities. During data collection, LiDAR point clouds, MC image frames, and IMU acceleration and angular velocity are stored in separate files. These raw data will be used in the next phase for trajectory reconstruction and fusion.

3.3 Simulation Part II: First Layer of the Navigation Filter - Data Processing

In the first filtering layer, raw data from IMU, MC, and LiDAR are processed independently to obtain pose estimates. Each sensor receives raw data from the DIM and computes its own estimate without cross-referencing others.

The IMU data, composed of accelerometer and gyroscope measurements, are used to reconstruct the rover's trajectory. Angular velocity is integrated to estimate yaw, which is then synchronized with MC timestamps. Linear acceleration is transformed into the ARF using the estimated heading, and integrated to retrieve x and y positions. The inputs to this process are the raw IMU measurements (linear acceleration and angular velocity) acquired in the rover's body frame. The output is a pose trajectory in the form (x, y, θ) referred to the ARF, obtained through integration without the aid of external references.

For the MC, a modified Oriented FAST and Rotated BRIEF SLAM (ORB-SLAM) extracts keyframes and builds a sparse 3D map. ORB features are matched across frames, and the resulting poses are refined via bundle adjustment. The estimated trajectory is scaled, rotated, and translated to align with ground truth and corrected for the MC offset to obtain rover poses in ARF. In this case, the input consists of

[1] https://www.flir.it/products/blackfly-s-usb3/?model=BFS-U3-13Y3C-C&vertical=machine+vision&segment=iis

the camera image frames and intrinsic parameters. The output is a corrected sequence of rover poses in the ARF, expressed as (x, y, θ), derived from the estimated MC trajectory and refined through extrinsic calibration.

Regarding the LiDAR, point clouds are preprocessed by removing ground planes and outliers. Key Lidar Odometry and Mapping (LOAM) features (edges and planes) are extracted and downsampled. A LOAM-based odometry system estimates the LiDAR pose. Since the LiDAR is rigidly attached to the rover, its yaw is taken as the rover's one. Map points and trajectory data are retrieved in the ARF. The input is the raw point cloud stream from the sensor. The output includes both the estimated rover trajectory in the ARF (x, y, θ) together with a preliminary 3D map of the environment, that will be used in later stages for mapping refinement.

The next paragraphs evaluate each sensor's performance in terms of accuracy, noise, and completeness of the reconstructed trajectory.

3.3.1 IMU Data Processing Results

The IMU, operating solely within the rover's body coordinates, provides direct motion data without reference to the external environment. To emulate realistic sensor behavior, different types of noise are introduced into the IMU measurements. These include white noise, representing random measurement error, and a constant bias applied to each axis as detailed in Tab. 1. Bias instability (drift) naturally emerges as the integration of these noisy signals over time.

Table 1 IMU noise parameters

Accelerometer	
Constant bias (m/s^2)	(0.08, -0.05, 0.03)
Noise power ($(m/s^2)^2/Hz$)	(0.0015, 0.0015, 0.0015)
Gyroscope	
Constant bias (rad/s)	(0.002, -0.002, 0.001)
Noise power ($(rad/s)^2/Hz$)	(0.00015, 0.00015, 0.00015)

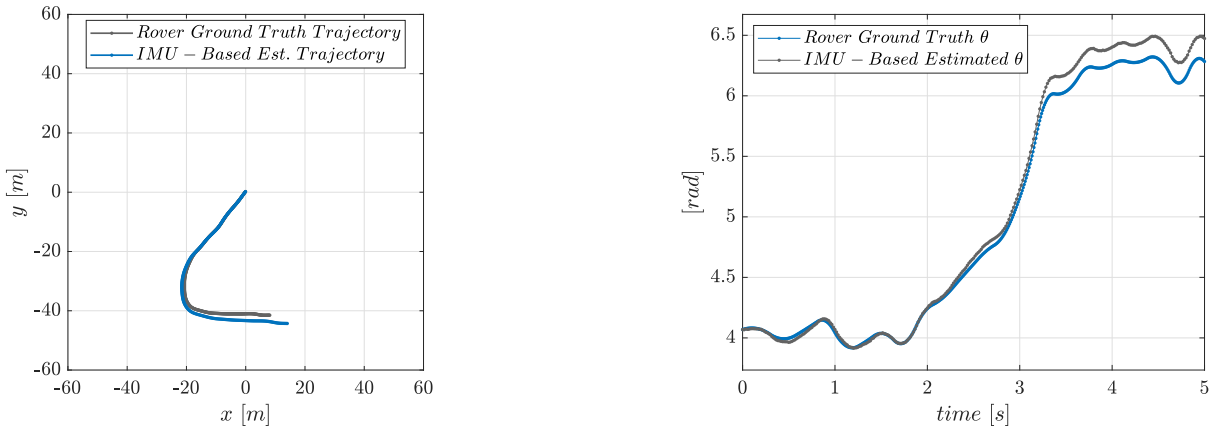
The resulting trajectory and heading angle estimations highlight the limitations of using IMU alone, reinforcing the need for sensor fusion due to visible drift over time. This behavior is illustrated in Fig. 3, showing the reconstructed poses deviating progressively from the ground truth.

3.3.2 MC Data Processing Results

The MC data are initially processed through the ORB-SLAM pipeline, producing a trajectory in the sensor's reference frame, lacking absolute scale. A transformation aligns it with the WCS and later with the ARF, accounting for the camera's mounting offset. The results show limited drift and reasonable accuracy in both position and heading, confirming the MC's reliability when supported by proper calibration and post-processing. The estimated pose evolution, compared against the ground truth, is displayed in Fig. 4.

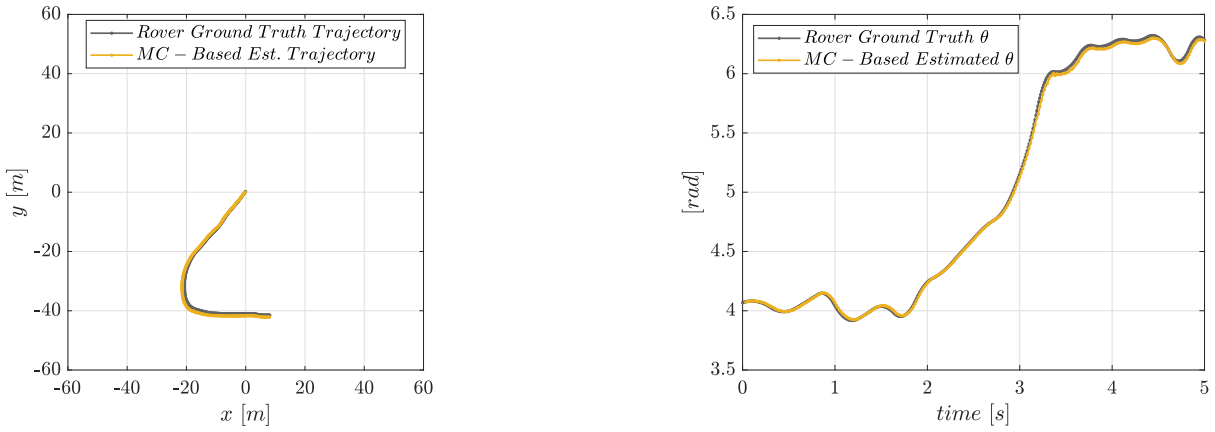
3.3.3 LiDAR Data Processing Results

The LiDAR proves to be the most accurate sensor in both localization and mapping. Following preprocessing steps such as ground plane removal and segmentation, the LOAM pipeline reconstructs



(a) **(b)**

Fig. 3 Final rover poses reconstruction using IMU data: (a) Trajectory, (b) Heading



(a) **(b)**

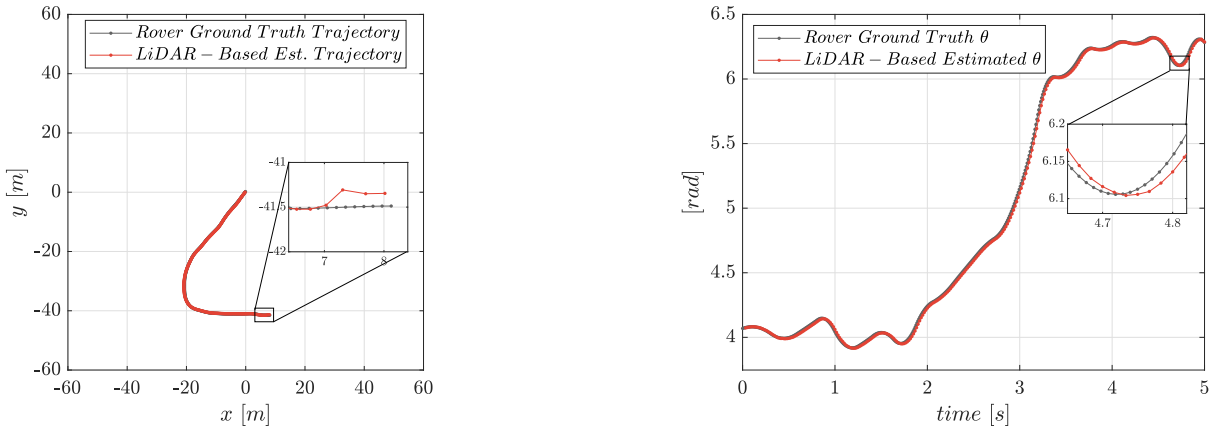
Fig. 4 Final rover poses reconstruction using MC data: (a) Trajectory, (b) Heading

the rover's trajectory and a dense environmental map. After correcting for mounting offsets, the LiDAR-based trajectory is transformed into the ARF and exhibits minimal deviation from ground truth, making it an essential component of the SLAM pipeline. The final map, after reference alignment, offers a highly accurate spatial reconstruction. Fig. 5 confirms the high accuracy of the reconstructed poses, closely following the simulated reference trajectory.

3.4 Simulation Part III: Second Layer of the Navigation Filter - Sensor Fusion

This stage fuses the pose estimates produced by individual sensors into a single, optimized trajectory through a two-step process.

The first loop employs a modified GB method that, unlike traditional approaches such as GraphSLAM, does not incorporate landmarks. Rather, it relies on independent trajectory estimates from the IMU and the MC, which are fused into a unified representation. This independence from mapped features and explicit control inputs makes the approach particularly advantageous for environments where landmark-based mapping is impractical or unreliable. The fusion strategy is implemented locally at each time step by combining the independent pose estimates from the two sensors through a weighted averaging process. Confidence matrices are assigned to each sensor and define their relative contribution to the



(a) **(b)**

Fig. 5 Final rover poses reconstruction using LiDAR data: (a) Trajectory, (b) Heading

fused output. Specifically, if both estimates are available, the optimized pose $\hat{\boldsymbol{p}}$ is computed as:

$$\hat{\boldsymbol{p}} = \frac{\mathbf{W}_{\text{IMU}} \cdot \boldsymbol{p}_{\text{IMU}} + \mathbf{W}_{\text{MC}} \cdot \boldsymbol{p}_{\text{MC}}}{\mathbf{W}_{\text{IMU}} + \mathbf{W}_{\text{MC}}} \tag{2}$$

where $\boldsymbol{p}_{\text{IMU}}$ and $\boldsymbol{p}_{\text{MC}}$ are the pose estimates from each sensor, and $\mathbf{W}_{\text{IMU}}$, $\mathbf{W}_{\text{MC}}$ are the associated confidence matrices. When only a single estimate from one of the two sensors is available, its corresponding output is used to ensure continuity.

The method can be classified as graph-based due to its structure and refinement approach. The trajectory is modeled as a graph, where each node represents a pose estimate, and edges encode constraints derived from the IMU and the MC. These constraints define the relative motion between consecutive poses and are incorporated into a weighted averaging process. This formulation ensures that the optimized pose remains closer to the more reliable sensor's estimate while still incorporating the complementary data from the other. Mathematically, this process minimizes the weighted sum of residuals, i.e., the difference between the optimized pose and each sensor estimate. Although the refinement occurs locally for each pose rather than globally across the entire trajectory, the method adheres to the graph-based principle of enforcing constraints between connected nodes. The resulting trajectory reflects the strengths of both sources and serves as input for the EKF refinement loop.

The second loop refines the optimized trajectory from the GB method using an EKF. The latter operates on a reduced state vector $\boldsymbol{p} = (x, y, \theta)$ and uses LiDAR measurements to correct the GB estimates. The EKF initialization phase involves defining the state covariance matrix P, the process noise covariance Q, and the measurement noise covariance R. The LiDAR pose estimates are temporally aligned with the GB trajectory to ensure consistent data fusion at each time step. When no LiDAR measurement is available, the state is propagated using the GB trajectory, and the uncertainty is increased to reflect model assumptions by adding Q to the current P. Once LiDAR data becomes available, a complete prediction-update cycle is performed. The predicted state and its covariance are initialized as:

$$\boldsymbol{p}_{\text{pred}} = \hat{\boldsymbol{p}}, \qquad P_{\text{pred}} = P + Q \tag{3}$$

The update step integrates the LiDAR measurement $\boldsymbol{z}$ using a direct observation model $H = I$. The Kalman gain is computed as:

$$K = P_{\text{pred}} H^{\top} (H P_{\text{pred}} H^{\top} + R)^{-1} \tag{4}$$

The state is then updated by correcting the prediction:

$$\boldsymbol{p}_{\text{est}} = \boldsymbol{p}_{\text{pred}} + K\left(\boldsymbol{z} - H\boldsymbol{p}_{\text{pred}}\right) \tag{5}$$

and the posterior covariance becomes:

$$P = (I - KH)P_{\text{pred}} \tag{6}$$

Unlike traditional EKF-based SLAM, which jointly estimates both poses and landmarks, this implementation simplifies the state, excluding explicit landmark tracking. Instead, LiDAR measurements are directly used for positional updates, reducing reliance on a control model. This approach leverages the high precision of LiDAR, which consistently outperforms IMU and MC in measurement accuracy. The prediction step uses the GB trajectory to propagate the rover's state forward. The update step refines this estimate by incorporating LiDAR measurements through the Kalman gain. This reduces dependency on odometry-based motion models, which are often prone to drift and require control inputs such as wheel velocities.

A final error analysis, regarding both loops, is conducted in the RRF, computing deviations in along-track s, cross-track n, and heading θ, with corresponding 3σ confidence bounds. These metrics summarize the localization accuracy achieved after both fusion steps, whose effectiveness is further evaluated in the following sections.

3.4.1 Graph-Based Method Results

The GB method fuses IMU and MC pose estimates, balancing their contributions through confidence-based weighted averaging. The resulting trajectory aligns closely with the MC estimate while reducing cumulative drift thanks to the IMU's high-frequency updates. As illustrated in Fig. 6, the positional and heading errors in the RRF are significantly lower than those from individual sensors, confirming the method's effectiveness.

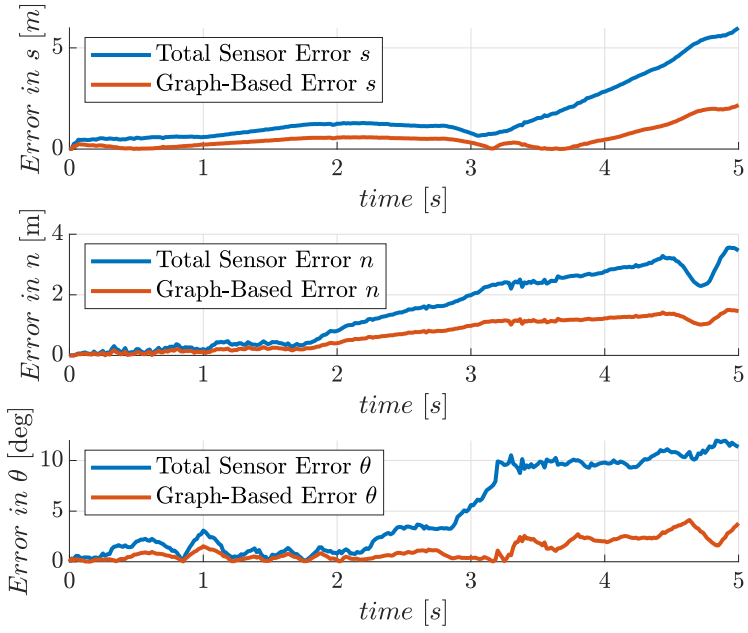
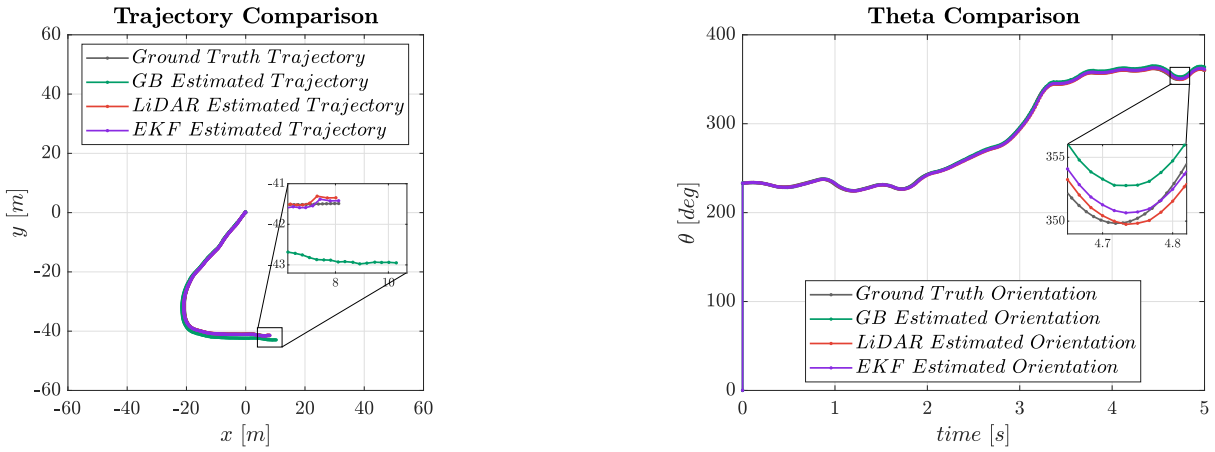


Fig. 6 Comparison between GB error and cumulative sensor errors in RRF

3.4.2 Extended Kalman Filter Results

The EKF further refines the GB trajectory by integrating LiDAR pose estimates. It operates on a reduced state and does not require control inputs. The choice of the EKF's noise covariance matrices was guided by the sensitivity analysis discussed later in Section 3.6, aiming to balance prediction responsiveness and measurement reliability. In particular, the measurement noise configuration reflects the expected performance of the LiDAR sensor, while the process noise ensures adaptability without overfitting. Fig. 7 illustrates the EKF's final trajectory and heading estimation. These results confirm the filter's effectiveness in smoothing the pose reconstruction and maintaining temporal consistency in a GPS-denied setting. The accuracy achieved through this refinement process will be further examined in the following section.



(a) **(b)**

Fig. 7 EKF results: (a) Trajectory estimation, (b) Heading estimation

3.5 Comparative Analysis

To evaluate the benefits of replacing control-based prediction with the GB trajectory, a comparison is made between the proposed method (EKF-GB) and a standard EKF implementation using odometry inputs (EKF-Odometry). Both filters share the same initialization parameters, ensuring that the only difference lies in the prediction model.

In EKF-Odometry, the prediction step uses noisy control inputs, which alone would result in a significant deviation from the true trajectory. Although LiDAR updates partially correct the trajectory, cumulative errors in heading and lateral displacement often remain. By contrast, the EKF-GB method leverages a fused IMU-MC trajectory as prior, which provides a more stable and accurate basis for prediction. Tab. 2 summarizes the average errors obtained for both configurations. The EKF-GB approach consistently outperforms its odometry-based counterpart across all components of the pose, with reductions of approximately 4% in along-track error, over 25% in cross-track error, and more than 20% in heading error. These results highlight the value of adding a lightweight graph-based layer before the EKF, especially when odometry is unreliable. The reduced errors suggest improved trajectory consistency and better alignment with the ground truth path, making the EKF-GB approach particularly suitable for long-term navigation in unstructured or GPS-denied environments.

While the observed gains might appear moderate, it is important to consider the context. Due to the high computational load imposed by the simulation framework, especially the integration of Unreal Engine™ and the detailed sensor modeling, tests were necessarily limited to short time spans. Nonetheless, the EKF-GB method demonstrated measurable improvements. It is reasonable to expect

Table 2 Mean errors comparison

Mean error	EKF-GB	EKF-Odometry
$\overline{\mathbf{err}}_s$ **(m)**	0.2920	0.3041
$\overline{\mathbf{err}}_n$ **(m)**	0.0414	0.0558
$\overline{\mathbf{err}}_\theta$ **(deg)**	0.6245	0.8005

that, over longer mission durations, the benefits of the proposed approach would become even more pronounced.

3.6 EKF Dependence on Noise Covariance Matrices

The impact of Q and R was also investigated. Lower values of Q reduce corrections from LiDAR, making the EKF overly reliant on prediction, while higher values allow more effective integration of measurements. Conversely, low R values overemphasize sensor data, which can be detrimental if measurements are noisy, while high R values reduce the filter's responsiveness.

The evolution of the 3σ uncertainty bounds shows that a balanced setting leads to rapid convergence and long-term consistency. Setting $Q = 0$ slows convergence and may cause the filter to exceed expected error thresholds. These findings suggest the potential of an adaptive tuning strategy for R based on residuals analysis. Such a solution could improve robustness and adaptability, enhancing performance.

4 Conclusions

This study presented a modular SLAM framework for lunar UGVs operating in GPS-denied environments, developed within a flexible simulation architecture integrating multiple software platforms and heterogeneous sensors. This setup enabled realistic development and validation of SLAM algorithms for lunar exploration.

Results demonstrate the effectiveness of the approach, with accurate trajectory reconstruction and error metrics consistently within 3σ bounds. Compared to standard EKF odometry methods, the proposed graph-based (GB) formulation achieved superior accuracy and stability. The system's independence from control inputs, odometry models, and global landmarks enhances robustness against drift and feature scarcity. Combined with its modularity, the EKF-GB framework offers a versatile, resilient alternative adaptable to evolving mission needs.

Future work includes adaptive tuning of noise covariances, exploring alternative filters like the Unscented Kalman Filter, enhancing simulation realism with varied terrains and dynamic obstacles, incorporating additional sensors (e.g., radar, thermal cameras), and extending to multi-robot systems to improve mapping and operational resilience.

These directions will further strengthen the framework as a robust foundation for autonomous navigation in complex, unstructured environments.

Acknowledgments

This work is largely inspired by the author's Master's Thesis [21] and further developed within the research framework established at Politecnico di Milano.

The author wishes to express sincere gratitude to Professor Mauro Massari, supervisor at Politecnico di Milano, for his invaluable guidance and constant support, providing the knowledge to tackle complex problems and fostering skills that proved essential in the development of this work. Special thanks are also extended to co-supervisor Annachiara Ippolito, whose knowledge, availability, and encouragement greatly contributed to the progress of this research.

Declaration of Use of Artificial Intelligence

Artificial intelligence was used exclusively to assist with proofreading, translation, and linguistic reorganization of the manuscript. The authors wrote the original text and provided the complete scientific content, including research design, data analysis, coding, and manuscript structure. The use of AI did not contribute to the generation of scientific results, images, or any substantive content. The authors remain fully responsible for the integrity of the work.

References

[1] Ronald Greeley. Lava tubes and channels in the lunar Marius Hills. *The Moon*, 3(3):289–314, Dec. 1971. doi: 10.1007/BF00561842.

[2] Raymond P. Martin and Haym Benaroya. Pressurized lunar lava tubes for habitation. *Acta Astronautica*, 204:157–174, 2023. doi: 10.1016/j.actaastro.2022.12.013.

[3] Xiaohang Qiu and Chunyu Ding. Radar observation of the lava tubes on the moon and mars. *Remote Sensing*, 15(11), 2023. doi: 10.3390/rs15112850.

[4] Audai K. Theinat, Anahita Modiriasari, Antonio Bobet, H. Jay Melosh, Shirley J. Dyke, Julio Ramirez, Amin Maghareh, and Daniel Gomez. Lunar lava tubes: Morphology to structural stability. *Icarus*, 338:113442, 2020. doi: 10.1016/j.icarus.2019.113442.

[5] Junichi Haruyama, Kazuyuki Hioki, Motomaro Shirao, Tomokatsu Morota, Harald Hiesinger, Carolyn H. van der Bogert, Hideaki Miyamoto, Akira Iwasaki, Yasuhiro Yokota, Makiko Ohtake, Tsuneo Matsunaga, Seiichi Hara, Shunsuke Nakanotani, and Carle M. Pieters. Possible lunar lava tube skylight observed by SELENE cameras. *Geophysical Research Letters*, 36(21), 2009. doi: 10.1029/2009GL040635.

[6] William Whittaker. Technologies enabling exploration of skylights, lava tubes and caves. In *NASA Innovative Advanced Concepts (NIAC) Phase*, 2012. `https://api.semanticscholar.org/CorpusID:52206458`.

[7] Filip Klaesson, Petter Nilsson, Tiago Stegun Vaquero, Scott Tepsuporn, Aaron Ames, and Richard M. Murray. Planning and optimization for multi-robot planetary cave exploration under intermittent connectivity constraints. In *Proceedings (venue unspecified)*, 2020. `https://api.semanticscholar.org/CorpusID:231797539`.

[8] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, USA, 2005. ISBN: 978-0-262-20162-9.

[9] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *IEEE Robotics & Automation Magazine*, 13(1):99–110, Jan. 2006. doi: 10.1109/MRA.2006.1638022.

[10] Cyrill Stachniss, John J. Leonard, and Sebastian Thrun. Simultaneous localization and mapping. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 1153–1176. Springer, Cham, Switzerland, 2016. doi: 10.1007/978-3-319-32552-1_46.

[11] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer, New York, NY, USA, 1990. doi: 10.1007/978-1-4613-8997-2_14.

[12] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, Dec. 2016. doi: 10.1109/TRO.2016.2624754.

[13] Rathin Chandra Shit. Precise localization for achieving next-generation autonomous navigation: State-of-the-art, taxonomy and future prospects. *Computer Communications*, 160:351–374, 2020. doi: 10.1016/j.comcom.2020.06.007.

[14] Jeffrey Martz, Wesam Al-Sabban, and Ryan N. Smith. Survey of unmanned subterranean exploration, navigation, and localisation. *IET Cyber-Syst. Robot.*, 2(1):1–13, 2020. doi: 10.1049/iet-csr.2019.0043.

[15] Jun Zhu, Hongyi Li, and Tao Zhang. Camera, LiDAR, and IMU based multi-sensor fusion SLAM: A survey. *Tsinghua Science and Technology*, 29(2):415–429, 2024. doi: 10.26599/TST.2023.9010010.

[16] Héctor Azpúrua, Maíra Saboia, Gustavo M. Freitas, Lillian Clark, Ali-akbar Agha-mohammadi, Gustavo Pessin, Mario F. M. Campos, and Douglas G. Macharet. A survey on the autonomous exploration of confined subterranean spaces: Perspectives from real-world and industrial robotic deployments. *Robotics and Autonomous Systems*, 160:104304, 2023. doi: 10.1016/j.robot.2022.104304.

[17] Chris Dinelli, John Racette, Mario Escarcega, Simon Lotero, Jeffrey Gordon, James Montoya, Chase Dunaway, Vasileios Androulakis, Hassan Khaniani, Sihua Shao, Pedram Roghanchi, and Mostafa Hassanalian. Configurations and applications of multi-agent hybrid drone/unmanned ground vehicle for underground environments: A review. *Drones*, 7(2), 2023. doi: 10.3390/drones7020136.

[18] Weifeng Chen, Chengjun Zhou, Guangtao Shang, Xiyang Wang, Zhenxiong Li, Chonghui Xu, and Kai Hu. Slam overview: From single sensor to heterogeneous fusion. *Remote Sensing*, 14(23), 2022. doi: 10.3390/rs14236033.

[19] Automated driving toolbox, 2025. Online; accessed 2025. `https://it.mathworks.com/products/automated-driving.html`.

[20] Vehicle dynamics blockset, 2025. Online; accessed 2025. `https://it.mathworks.com/products/vehicle-dynamics.html`.

[21] Niccolò Goretti. Enhanced sensor fusion for SLAM-driven autonomous navigation of lunar UGVs in GPS-denied environments. Master's thesis, Politecnico di Milano, Dept. of Aerospace Science and Technology, Milan, Italy, 2024.