drones

Article

uVGS-2: The Micro Video Guidance Sensor: A 6-DoF Robust Pose Estimator for Autonomous Proximity Maneuvers in Drones, Spacecraft and Mobile Robot Navigation

Hector Gutierrez [1,*], Jose Cornejo [1] and Ivan Bertaska [2]

[1] Mechanical and Aerospace Engineering, Florida Institute of Technology, Melbourne, FL 32901, USA; jcornejoaguilar@my.fit.edu

[2] Control Systems Design and Analysis Branch, EV-41, NASA Marshall Space Flight Center, Huntsville, AL 35812, USA; ivan.r.bertaska@nasa.gov

* Correspondence: hgutier@fit.edu; Tel.: +1-(321)-674-7321

Highlights

What are the main findings?

- uVGS-2 provides accurate six-degrees-of-freedom pose estimation in drone and robotic platforms by photogrammetric processing of known beacon geometry, achieving centimeter-level position accuracy and sub-degree attitude estimation errors in real-time, under varying lighting conditions.
- uVGS-2, in sensor fusion with the onboard localization system, has been experimentally demonstrated in NASA's Astrobee free-flying robot, with real-time deployment using the camera and CPU of the host robot. Improved robustness and accuracy are achieved compared to Astrobee's native map-based localization system.

What are the implications of the main findings?

- uVGS-2 enables 6-DOF pose estimation in drones, spacecraft, and mobile robots, supporting proximity operations such as rendezvous, docking, formation flight, precision landing in GPS-denied environments, and re-localization when used in fusion with other localization methods.
- The sensor fusion of uVGS-2 with the onboard state estimation enhances robustness, accuracy and scalability. uVGS-2, developed as a ROS node, enables deployment across platforms while eliminating dependency on external positioning systems.

Abstract

This paper presents the Micro Video Guidance Sensor Version 2 (uVGS-2), a ROS-based vision navigation framework for real-time six-degrees-of-freedom pose estimation in drones, spacecraft, and autonomous robotic platforms operating in GNSS-denied environments. The system evolves from the previous Smartphone Video Guidance Sensor (SVGS) architecture through a modular C++ implementation, including advanced image preprocessing, deterministic blob sorting, and an optimized perspective-4-point solver using a Lie-algebra-based analytical Jacobian formulation. The proposed architecture achieves computationally efficient photogrammetric state estimation using onboard camera and processor resources, enabling deployment in resource-constrained systems. Experimental validation was conducted in NASA's Astrobee free-flying robot, both at the International Space Station (ISS), for SVGS, and by ground testing through real-time sensor-fusion with Astrobee's graph-based localizer (Astroloc), for uVGS-2. Results demonstrate robust centimeter-level accuracy in relative position and attitude estimation under illumination

Academic Editor: Pablo Rodríguez-González

Received: 13 May 2026
Revised: 7 July 2026
Accepted: 8 July 2026
Published: 14 July 2026

disturbances, partial occlusions, and intermittent loss of line-of-sight. The framework can be used in robotic platforms and autonomous UAV operations, including precision landing, formation flight, and cooperative navigation in environments where GNSS signals are unavailable or intermittent.

Keywords: drones; vision-based 6-DOF pose estimation; autonomous proximity operations; photogrammetric state estimation; navigation in GPS-denied environments; robotics

1. Introduction

The Smartphone Video Guidance Sensor (SVGS) is a vision-based relative pose estimation system developed for autonomous proximity operations in aerospace robotic platforms. The system estimates the 6-DOF pose of a target in the camera's coordinate system using monocular images from a smartphone camera, enabling real-time determination of relative position and orientation without the need for ranging hardware [1–4]. SVGS relies on the detection of retroreflective or illuminated fiducial markers arranged in a predefined geometric pattern. The approach substantially reduces mass, power consumption, and integration complexity, making SVGS particularly suitable for resource-constrained applications such as CubeSats, free-flying robotic platforms [2], unmanned aerial vehicles [3], and autonomous mobile robots. Pose estimation is based on photogrammetric reconstruction, where images from the target markers are used to infer the relative spatial configuration between the camera and the target structure. Fiducial points in the image are processed as geometric features whose spatial relationships are known. The use of monocular vision simplifies hardware implementation while maintaining sufficient accuracy for proximity operations [4,5]. To improve detection and attitude estimation, the fiducial targets follow a tetrahedral geometry, which provides better performance than flat targets such as AR tags or Apriltags [6,7]. SVGS images are processed through thresholding and segmentation to isolate target blobs from the background environment, determine centroid locations and validate blob morphology prior to geometric correspondence analysis. The relative pose is estimated by a photogrammetric procedure that minimizes reprojection error [3,8]. The estimated 6-DOF pose can be used to support autonomous maneuvers and relative motion control in spacecraft [1–3], and has been successfully demonstrated in precision drone landing missions [9–11].

The main contributions of this paper to the state-of-the-art are as follows:

1. A vision-based pose estimation engine is proposed (uVGS-2) that mitigates geometric degeneracies in the solution of the P4P pose estimation problem and operates with low computational overhead on resource-constrained embedded platforms.
2. uVGS-2 can be deployed as a ROS node in drone and mobile robot applications, using hardware resources (camera and CPU) from the host drone or robot.
3. uVGS-2 has significantly better performance than existing vision-based pose estimation techniques (such as Aruco or Apritags) in the estimation of attitude angles, and is significantly more robust to variations of the local illumination conditions.
4. uVGS-2 can be used in conjunction with the host robot's localization scheme as part of a sensor fusion formulation, which mitigates the effect of beacon occlusions, loss of line of sight, and loss of beacon detections.
5. SVGS and uVGS-2 have been deployed and successfully demonstrated in NASA's free-flying Astrobee platform, in both the International Space Station (SVGS) and ground testing at NASA Ames' Granite Lab (uVGS-2).

6. Astrobee deployment demonstrated successful integration of uVGS-2 with Astrobee's multi-sensor factor-graph localization framework (AstroLoc), showing resilience against occlusions, loss of beacon line of sight, and loss of beacon detections.

Section 1 presents the experimental framework and results from deployment and testing of SVGS onboard the International Space Station (ISS). Section 2 presents the algorithmic and mathematical methodology of uVGS-2, including image preprocessing, blob extraction, on-manifold Lie-algebra optimization, and middleware integration. Section 3 presents the experimental framework and results from deployment, integration and testing of uVGS-2 in Astrobee at NASA Ames' Granite Lab. Sections 4 and 5 discuss the framework's evolution, capabilities, and applications across drone and mobile robot systems and outline operational limitations and directions for future research.

1.1. The SVGS-1 Mission on the International Space Station

The SVGS-1 mission was a technology demonstration to evaluate performance, robustness, and operational feasibility of SVGS when deployed in a representative space environment for proximity operations and formation flight [4,12], with SVGS deployed as a software sensor that uses hardware resources (camera and CPU) of a host robot platform such as NASA's Astrobee, using illuminated beacons as shown in Figures 1 and 2. The SVGS mission APK was implemented as a Java Android application, integrating the Astrobee API, guest science library and ground data system (GDS), as shown in Figures 3 and 4. Vision-based localization techniques such as those discussed in [13–18] showcase the fundamental differences between the proposed approach and existing methods described in the literature.

SVGS Maneuver 1 is shown in Figure 5. It was used to check SVGS functionality and parameter calibration; it included translation in both directions in an axis perpendicular to the JPM1F5 panel, followed by a sequence of rotations in yaw, pitch and roll, while acquiring both Astrobee metrology and SVGS 6-DOF pose measurements.

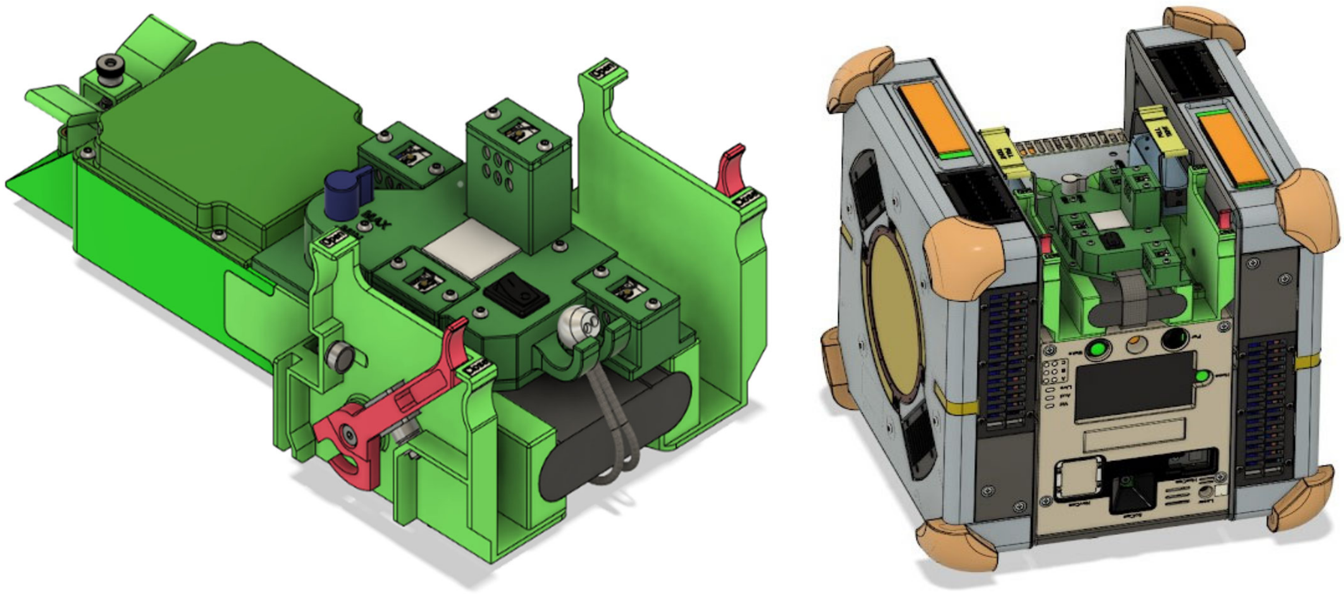

Figure 1. Integration of the SVGS beacon with NASA's Astrobee free-flying robot (Aurora Space Systems, Cambridge, MA, USA) using the M561 payload interface. The SVGS beacons can be mounted either on the robot payload bay or on ISS structural elements, enabling multiple experimental configurations. The ISS SVGS beacon was custom designed and build for integration with Astrobee.

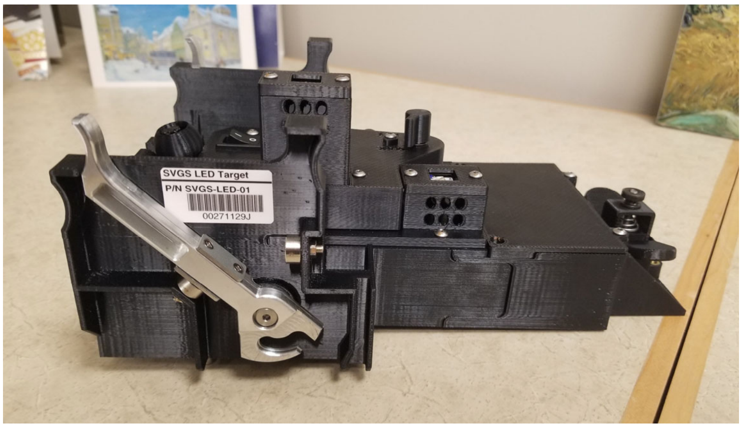

Figure 2. The flight-qualified SVGS tetrahedral beacon (Rev. 8) includes a DC–DC converter, LiPo battery, and overdischarge protection circuitry. A dimmer allows for the adjustment to different illumination conditions within the ISS environment.

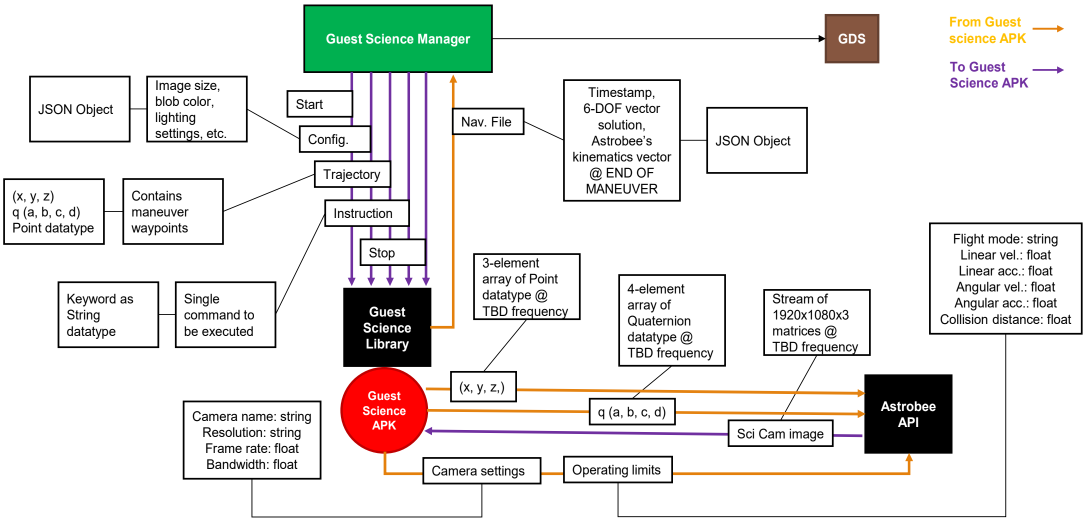

Figure 3. SVGS integration with Astrobee's guest science library and application programming interface (API). SVGS is initialized, executed, and monitored through service calls, conditional execution flags, and telemetry streaming. Synchronization constraints between asynchronous image processing and the deterministic control loop illustrate timing, thread management, and state validation. Pose data are transmitted to both onboard control modules and the ground data system (GDS).

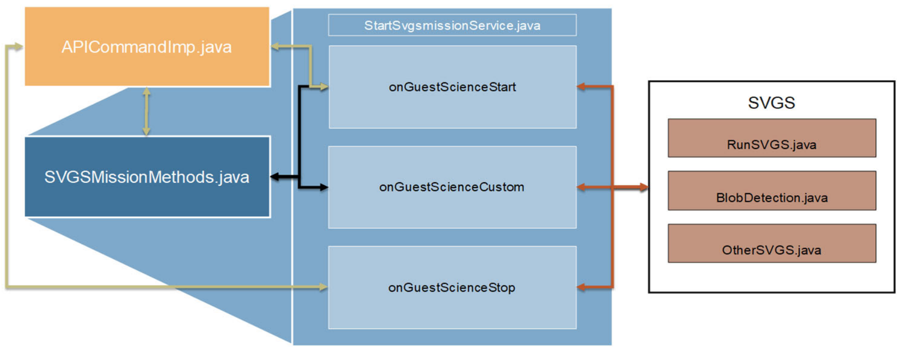

Figure 4. SVGS mission Android package (APK) architecture. The perception module (SVGS, **right**) includes the vision pipeline, while the control interface (**center**) manages communication with Astrobee's GNC subsystem. The telemetry module supports data logging and transmission to the ground data system, incorporating rate limiting to comply with bandwidth constraints.

Figure 5. SVGS Maneuver 1: Axial translations, axial approach and rotations in roll, pitch, and yaw.

In Maneuver 2 (Figure 6), the "leader" Astrobee performs a 6-DOF trajectory with respect to a wall-mounted target. The "follower" follows the leader in a parallel vertical plane, at a fixed distance. This imposes stricter requirements on estimation and synchronization, as errors in the leader's motion propagate into the follower's motion control. Maneuver 2 was divided into 2A (leader square maneuver using Astrobee metrology and SVGS), 2C (leader square maneuver using only Astrobee metrology) and 2F (leader–follower maneuver). Experimental results are shown in Figures 7–9. Close alignment between Astrobee metrology and SVGS measurements indicates strong temporal coherence and low estimation error across all axes. Motion commands (green traces) show correlation between commanded trajectories and measured response.

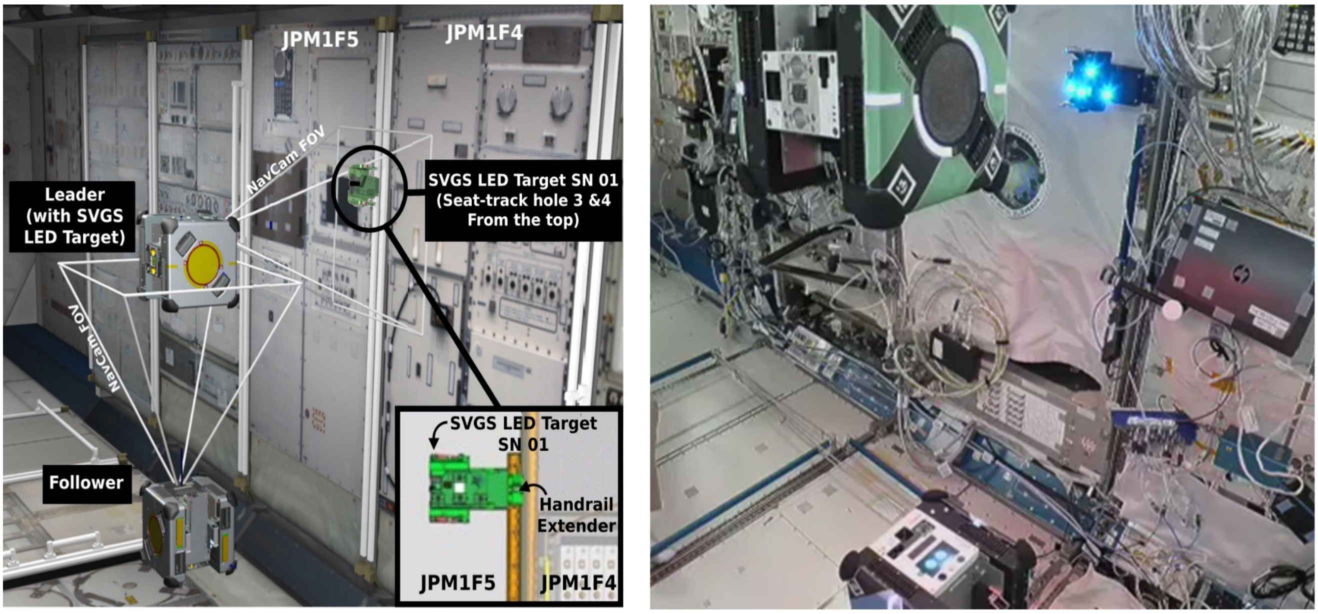

Figure 6. SVGS Maneuver 2: Leader–follower maneuver: two Astrobee robots operate in formation flight using SVGS-based relative navigation. The leader executes a predefined trajectory relative to a fixed beacon, while the follower maintains a constant separation distance by tracking the leader's motion.

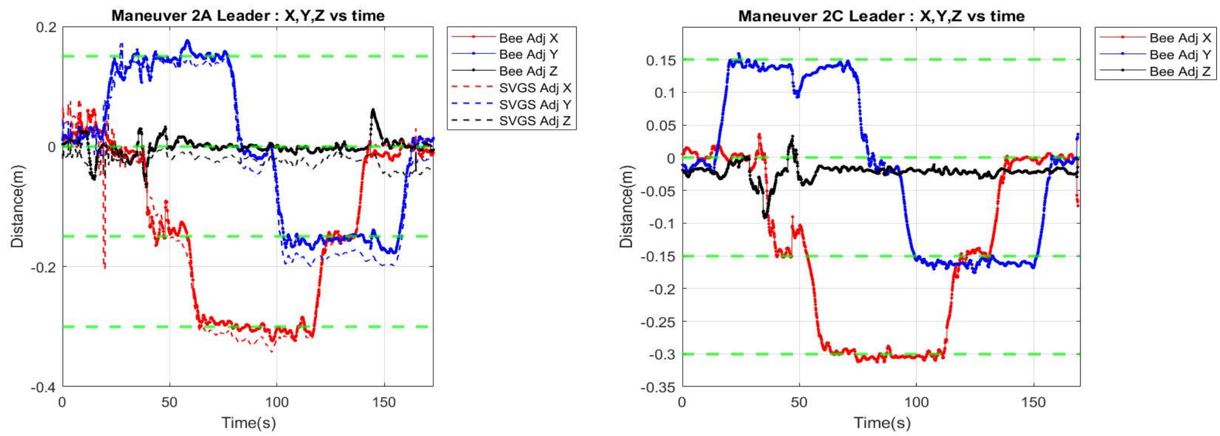

Figure 7. Maneuver 2A and 2C, SVGS Science 1. XYZ robot coordinates vs. time: Astrobee metrology (solid traces) and SVGS position estimates (dotted traces). Motion commands are shown in green.

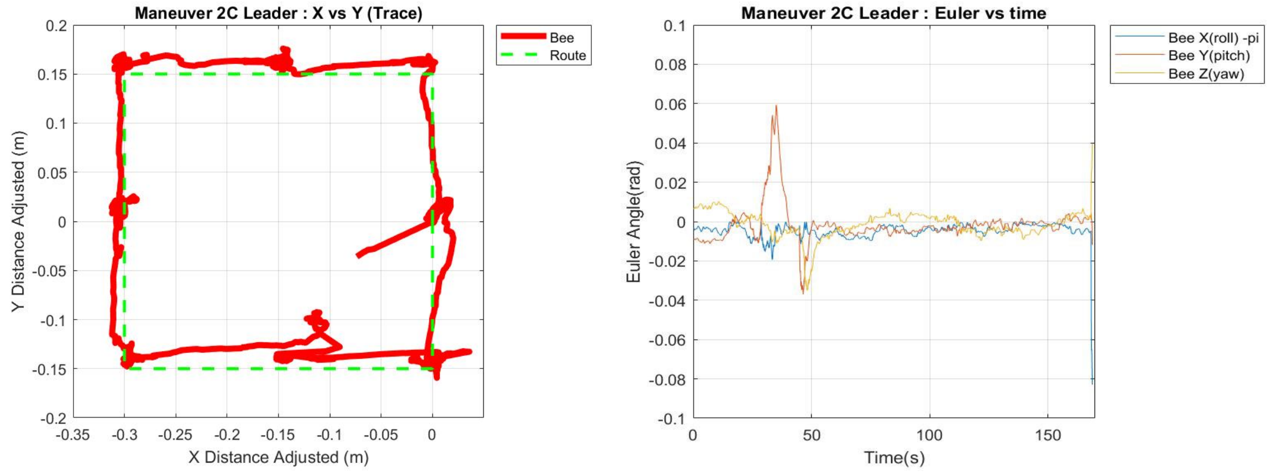

Figure 8. Maneuver 2C, SVGS Science 1. (**Left**): Motion command (green trace) and corresponding XY robot position (red); (**Right**) Astrobee Euler angles as function of time.

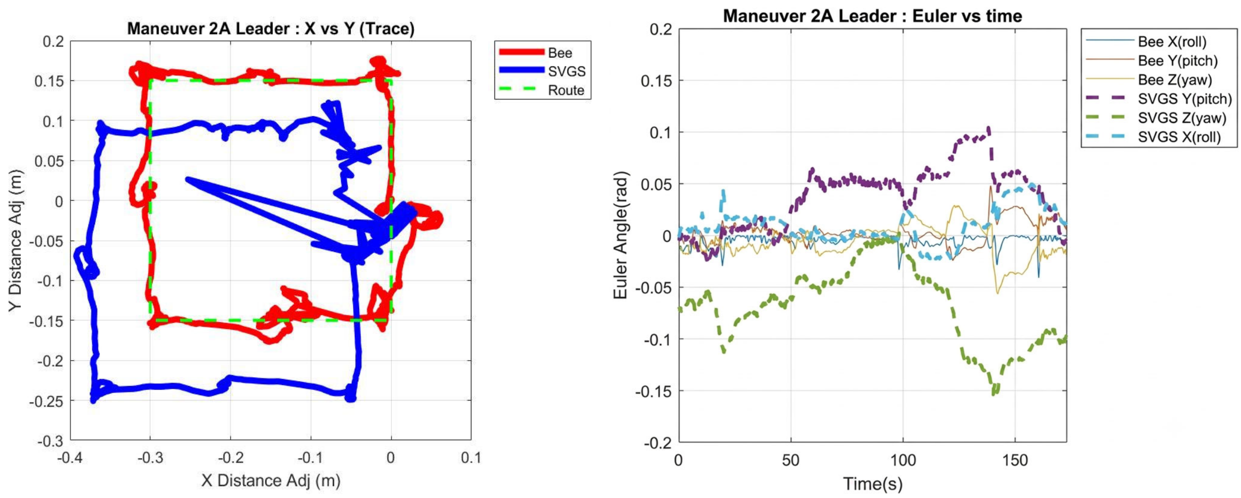

Figure 9. Maneuver 2A, SVGS Science 1. (**Left**) Motion command (green trace) and corresponding XY robot position (red), SVGS data are shown in blue. (**Right**) Astrobee Euler angles as function of time, by both local metrology (Bee) and SVGS.

SVGS Science 1 (24 June 2022) included two check maneuvers (2C and 2A) to verify correct functionality of the SVGS APK. SVGS Science Session 2 (6 July 2022) included a baseline repeat of 2C and 2A (Figure 10) plus the actual formation flight maneuver, Maneuver 2F

(Figure 11); time domain behavior of XYZ RPY coordinates is shown in Figures 12 and 13. Figure 11 shows leader and follower formation flight behavior; follower motion in the XY vertical plane tracks the leader's motion at a controlled offset, illustrating the effectiveness of SVGS to support relative navigation in formation flight at close range. This is further illustrated in Figures 12 and 13, where XYZ coordinates and Euler angles for both agents are plotted, illustrating that the follower maintains a stable relative configuration relative to the leader's trajectory. These results illustrate SVGS capability of supporting formation flight in proximity operations with partial occlusion of scene features.

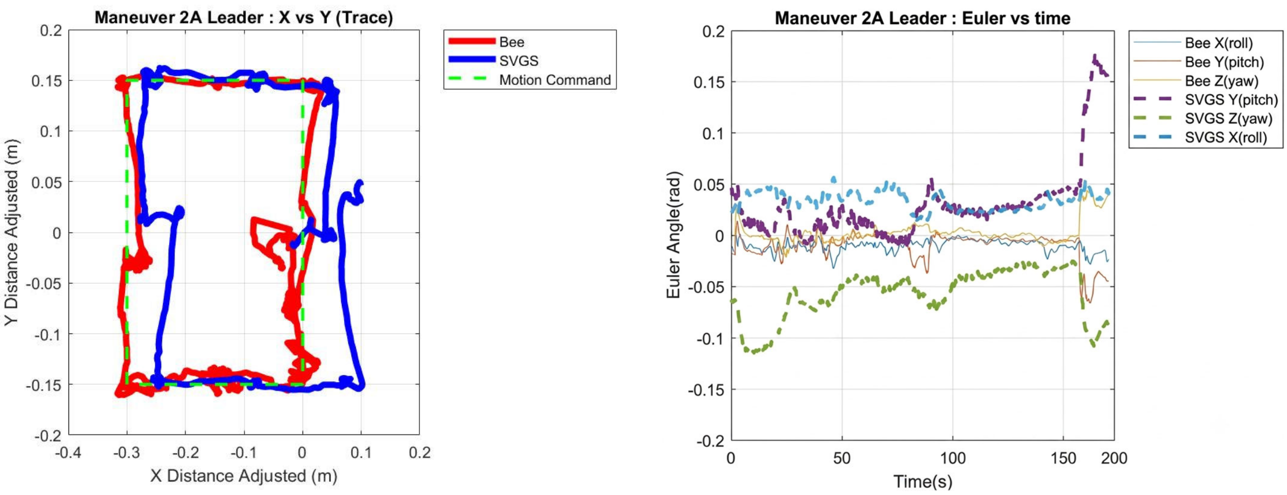

Figure 10. Maneuver 2A (Queen) + 2F (Bumble), SVGS Science 2: Formation flight: Leader results. SVGS follows Astrobee metrology with no significant drift. (**Left**) Motion command (green trace) and corresponding XY robot position (red), SVGS position data are shown in blue. (**Right**) Astrobee Euler angles. SVGS measurements are shown as dotted lines, Astrobee metrology as solid lines.

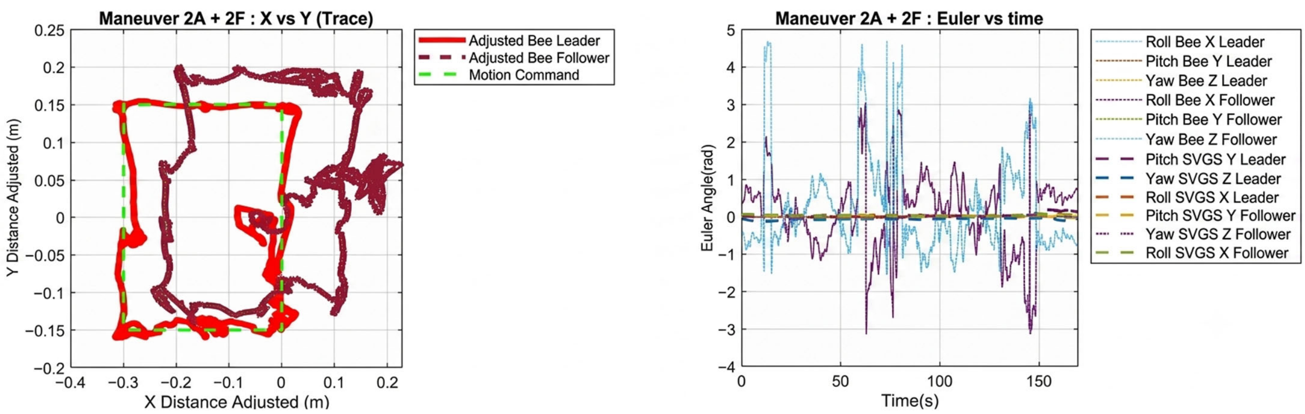

Figure 11. Maneuver 2A (Queen) + 2F (Bumble), SVGS Science 2: Formation flight. (**Left**) Leader's motion command (green trace) and leader position (red), follower position is shown in crimson. (**Right**) Astrobee Euler angles as function of time. Estimation of Euler angles by Astrobee native metrology is significantly affected in the follower due to partial blockage of the follower's line of sight (LOS) caused by the leader's proximity. This illustrates the advantage of having an independent pose estimation method such as SVGS in proximity operations when native localization is map-based and partial blockage of visual features is likely.

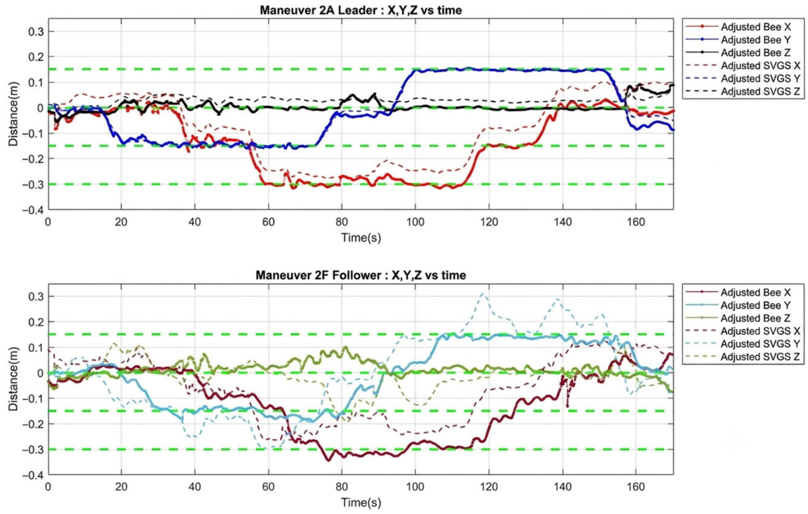

Figure 12. Maneuver 2A (Queen) + 2F (Bumble) during SVGS Science 2. XYZ robot coordinate vs. time. Astrobee coordinates are shown in solid lines, both for leader and follower, SVGS coordinates are shown in dotted lines, illustrating delay in follower motion. Green traces are command values.

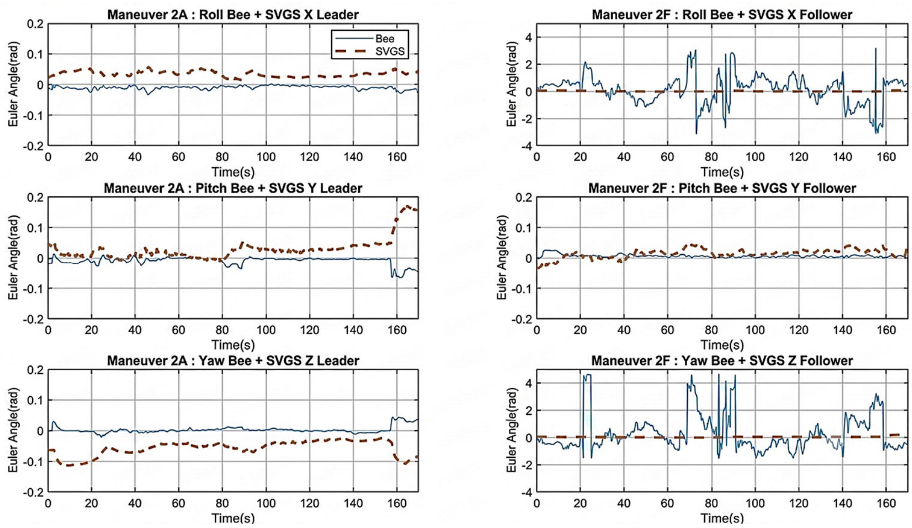

Figure 13. Maneuver 2A (Queen) + 2F (Bumble), SVGS Science 2. Euler angles, Astrobee estimates shown in solid lines, SVGS coordinates in dotted lines. Astrobee estimates of roll and yaw based on native Astrobee metrology are significantly affected in follower due to partial block of LOS by Leader. SVGS estimation of Euler angles remains correct despite the leader's proximity.

Maneuver 3 (SVGS Science Session 3) was a demonstration of SVGS range extension and handover capabilities using a multi-target path-following maneuver. The robot must sequentially navigate between four SVGS beacons distinguished by color and spatial arrangement, while moving on a plane parallel to ISS panel JPM1F5. The maneuver's configuration is shown in Figure 14, left, while Figure 14, right shows the exact spatial location of the SVGS beacons on the ISS JPM forward wall, where LED 1 and 4 are blue, LED 2 is red, and LED 3 is green. The maneuver requires the robot to move from an initial position facing the first target and proceed through a series of waypoints corresponding to subsequent targets, stopping at each location while maintaining stable pose. During Maneuver 3A (Figure 15), SVGS hue change is determined based on robot location. When the hue is switched, other beacons on the image stream are ignored. AstroBee moves in the YZ plane and the robot follows the prescribed path from beacon to beacon.

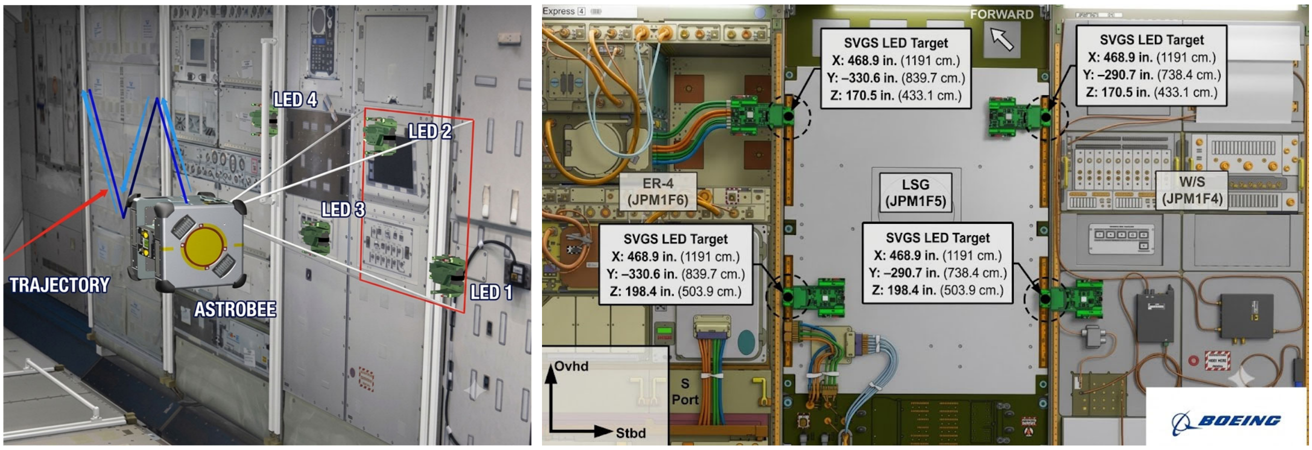

Figure 14. (**Left**) Maneuver 3: Multi-target path following maneuver, as the robot sequentially navigates between multiple SVGS targets on a vertical plane parallel to the ISS wall. The path includes transitions where SVGS must switch its visual reference from one beacon to the next. This requires robust target identification where more than one beacon shows in the image stream. (**Right**) SVGS beacon locations on ISS JPM forward wall, LED 1 and 4 are blue, LED 2 is red, and LED 3 is green.

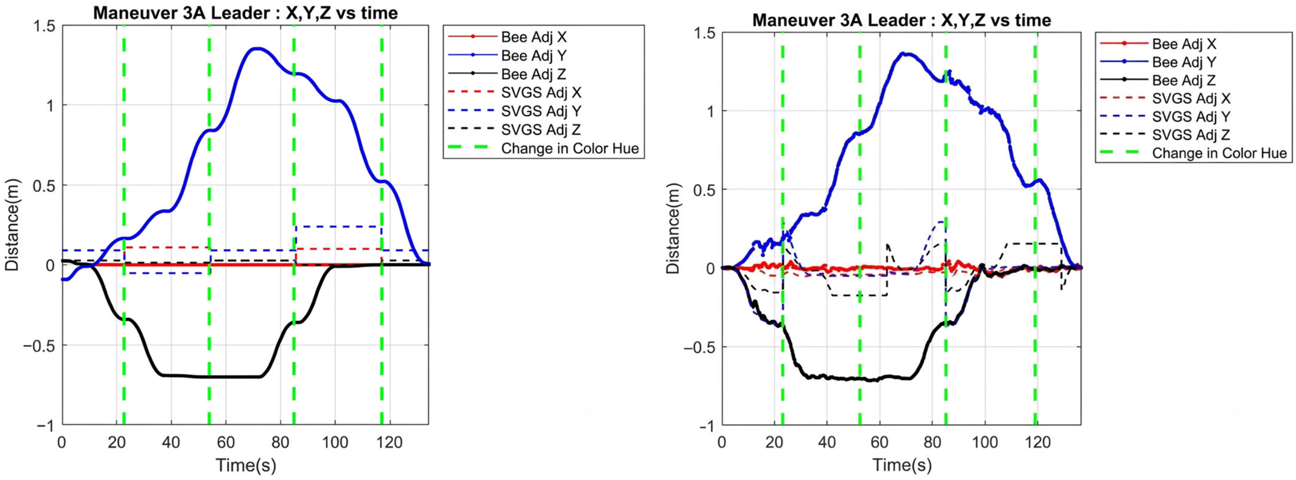

Figure 15. (**Left**). Hardware-in-the-loop (HIL) simulation of maneuver 3A using Gazebo and an Inforce 6640SBC single board computer. (**Right**) Maneuver 3A, during SVGS Science 3. XYZ robot coordinate vs. time are shown in solid trace (red, blue, black) while the corresponding SVGS readings are shown in dotted lines. Changes in color of interest are shown as vertical green traces. Close agreement between the HIL simulation and experimental data is shown throughout the maneuver.

Maneuver 3A was demonstrated and tested with hardware-in-the-loop (HIL) simulations using Rviz and Gazebo. For the HIL, the sequence of SVGS targets was presented to a camera mounted to an Inforce 6640SBC board. By manually approaching the targets to the camera line of sight, it was possible to mimic the sequence in which Astrobee would see the targets in the actual maneuver and record the corresponding robot trajectory.

Maneuver 3 comprised an error correction maneuver (Maneuver 3B) where SVGS readings are used to update motion commands as Astrobee moves between designated SVGS targets. Changes in color of interest (hue) are shown as step changes in SVGS coordinates (Figure 16, right). Results from the HIL simulation of Maneuver 3B are shown in Figure 17, left, and the corresponding robot trajectory in the YZ plane is shown in Figure 17, right. Figure 18 shows the results from Maneuver 3B.

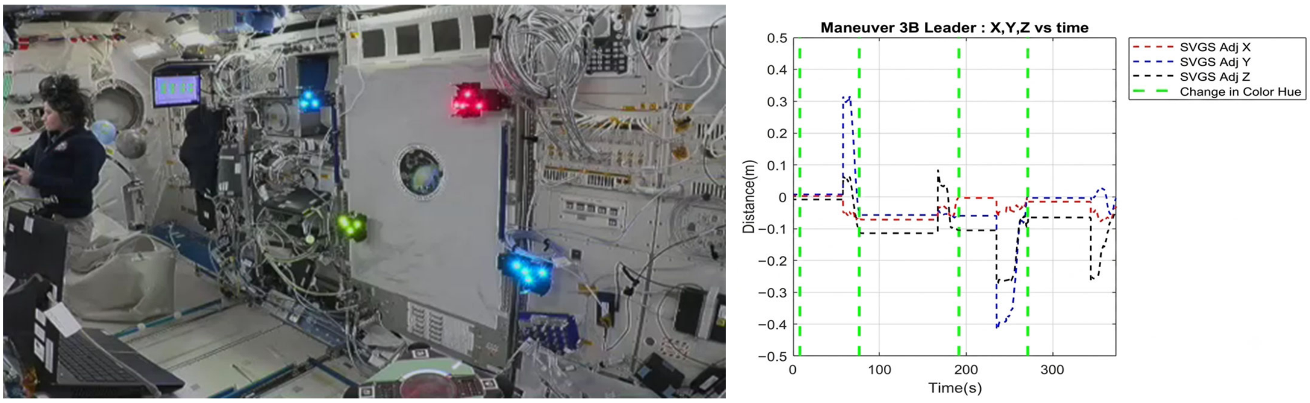

Figure 16. (**Left**) Maneuver 3B, SVGS Science Session 3. SVGS readings are used to update motion commands as Astrobee moves between designated SVGS targets. The arrangement demonstrates range extension in SVGS by using multiple beacons, which requires the ability to ignore multiple targets present on the image stream, and handover is handled based on different hues. (**Right**) SVGS output during Maneuver 3B. As Astrobee changes color of interest (hue), handovers are detected as step changes in SVGS coordinates. During transition between targets, SVGS measurements latch to the last valid value, which shows as horizontal flat lines.

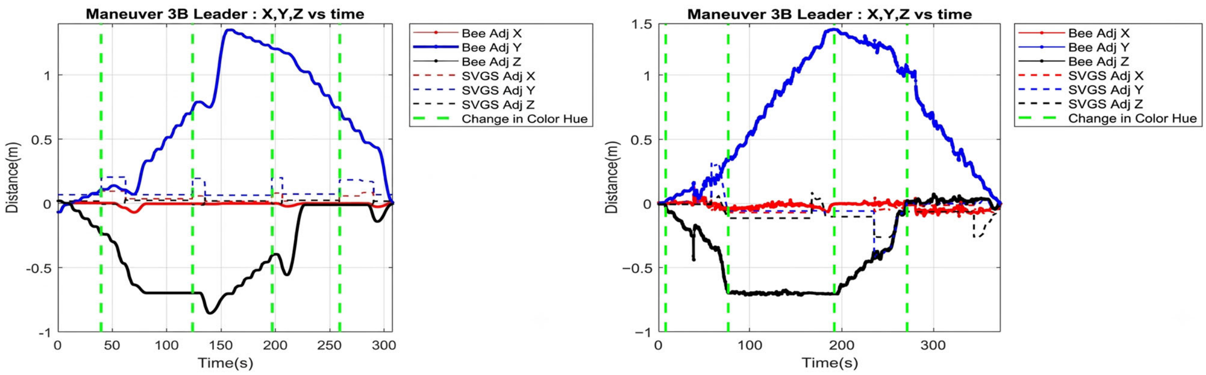

Figure 17. (**Left**) HIL simulation of Maneuver 3B. (**Right**) Maneuver 3B data, SVGS Science 3. XYZ robot coordinates are shown in solid trace (red, blue, black), SVGS readings are shown in dotted lines. Changes in hue of interest are shown as vertical green lines. Maneuver 3B demonstrates range extension with error correction based on SVGS readings and handover based on different hues.

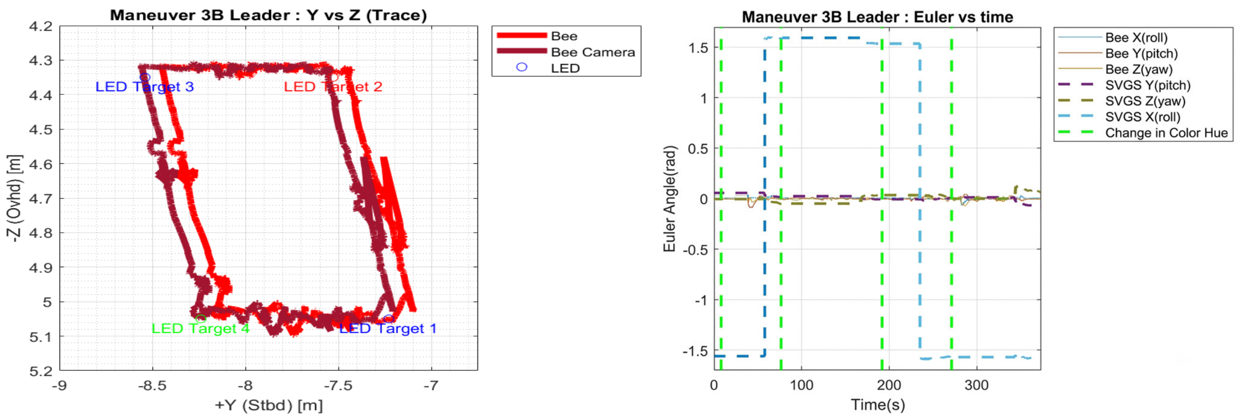

Figure 18. Maneuver 3B, SVGS Science 3. (**Left**) Robot trajectory on YZ plane. (**Right**) Changes in Euler angles as robot transitions from one target to the next. Green dotted lines show a change in color of interest. The trajectory in YZ highlights the system's ability to handle handover transitions.

SVGS Science 3 (09/20/2022) included two SVGS maneuvers (3A and 3B—Figure 18). HIL simulations were used to verify the correct functionality of the APK. The data collected during SVGS Science 3 demonstrates range extension and handover with SVGS. Maneuver 3B demonstrated the use of incremental motion commands for error-based navigation. Maneuver 4 (SVGS Science Session 4) combined multi-target navigation with leader–follower formation flight in a single maneuver, as shown in Figure 19. The leader robot navigates through the multi-target sequence, while the follower attempts to maintain a fixed relative position based on SVGS measurements. This configuration requires relative pose estimation under conditions where visual occlusions degrade the follower's localization performance.

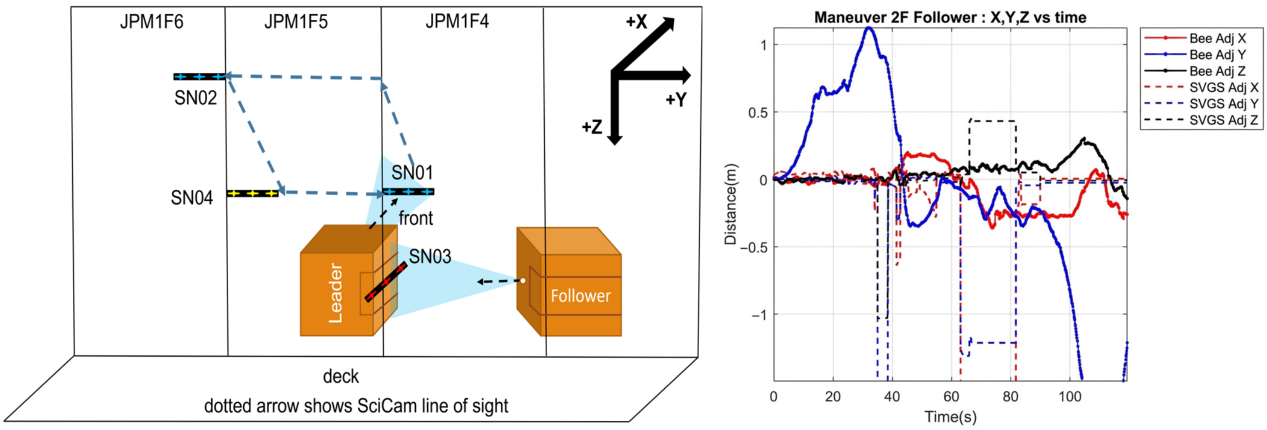

Figure 19. Maneuver 4: Multi-target leader–follower maneuver. (**Left**) The leader navigates relative to a set of SVGS targets of different colors while the follower follows the leader based on SVGS measurements. (**Right**) Maneuver 4A, SVGS Science 4. Follower robot coordinates (XYZ vs. time) are shown in solid trace (red, blue, black) while SVGS readings are in dotted lines. The plot illustrates a breakdown in the native Astrobee follower localization due to partial block of line of sight caused by leader proximity, showing the impact of poor feature availability and occlusions on localization and navigation performance during proximity operations.

Results from Maneuver 4 (Figure 19, left) reflect a breakdown in the native Astrobee localization framework on the follower robot, leading to a loss of navigation stability. The leader's location at the start of the maneuver leads to partial occlusion of the localizer's reference surface: the lack of sufficient visual features, combined with the proximity of the leader's beacon as a strong light source, leads to degraded performance in Astrobee's native localization system. As a result, the follower is unable to maintain stable motion, highlighting a critical limitation in the native AstroLoc to support formation flight.

1.2. uVGS-2: Micro Video Guidance Sensor Version 2.0

The evolution toward the Micro Video Guidance Sensor (uVGS-2) starts from the Advanced Video Guidance Sensor (AVGS) [19,20] developed at NASA's Marshall Space Flight Center for autonomous rendezvous and docking of spacecraft. AVGS provided vision-based estimation of the six-degrees-of-freedom (6-DOF) relative state vector between chaser and target vehicles [21]. The system relied on laser illumination to excite retroreflective target arrays mounted on the client spacecraft, producing high-contrast features captured by a radiation-tolerant CMOS imaging system. Relative pose was obtained through centroid extraction of illuminated markers followed by photogrammetric reconstruction and inverse perspective mapping. Flight demonstrations such as DART [19] and Orbital Express [21] validated AVGS in 6-DOF motion with variable illumination conditions, confirming its accuracy and operational robustness in proximity operations.

The Smartphone Video Guidance Sensor (SVGS) [1,3,4] emerged as a simplification of AVGS by replacing specialized sensing hardware with a smartphone, integrating camera, illumination, and embedded processing. It preserved the core photogrammetric framework while eliminating high-cost optical subsystems and enabling a software-defined sensing paradigm. SVGS, implemented in Java for Android, solves the perspective-4-point (P4P) problem to estimate the 6-DOF pose from monocular images. A modified “headless” SVGS variant was developed for the Astrobee platform, removing smartphone-specific interfaces and enabling direct use of host CPU and camera. The SVGS science sessions validated the feasibility of software-centric vision navigation using embedded platform assets. The Micro Video Guidance Sensor (uVGS) [22] represents a further shift towards a deterministic real-time implementation. Developed at NASA MSFC, uVGS was implemented in C to support real-time execution on FPGA-based systems [23]. Its modular library structure decouples algorithmic components from sensor and hardware interfaces, enabling portability across platforms. Validation was conducted primarily via synthetic images, making assessment independent of hardware.

uVGS-2 [12] is implemented in C++ as a robot operating system (ROS) node, enabling seamless interoperability within different camera systems, autonomy stacks, and robotic aerial, ground, and space platforms. uVGS-2 includes significantly expanded image pre-processing, using spatial decimation, adaptive intensity-based thresholding, Gaussian and median filtering, and intensity masking to improve signal isolation under non-uniform illumination in high-noise environments. Feature extraction is performed through contour-based blob detection combined with morphological refinement operations such as erosion and dilation to enhance centroid stability and eliminate spurious blob detections. Camera calibration is upgraded with lens distortion compensation to ensure geometric consistency in wide-field imaging systems. A geometry validation module enforces consistency of detected fiducial configurations through inter-point distance and angular constraints, improving robustness against false positives and partial occlusions. At the core of uVGS-2 lies an optimized perspective-4-point solver that replaces numerical differentiation with an analytical Jacobian formulation derived using Lie algebra [24,25], significantly improving convergence speed, numerical stability, and computational efficiency in the nonlinear optimization, leading to pose update rates that are up to six times faster than SVGS in the same CPU platform.

1.2.1. Technical Background and Initial Considerations of the uVGS-2 Mission

uVGS-2 was developed as a platform-independent, real-time 6-DOF pose estimation software sensor. Unlike the SVGS-1 mission, where SVGS functioned as an Android application for the guest science platform (HLP), the SVGS-2 mission sought integration at the mid-level processor (MLP), enabling direct interaction with Astrobee’s localization and control pipelines via ROS. In the SVGS-2 mission, uVGS-2 is a complementary pose estimation sensor for proximity operations via sensor fusion with Astrobee’s native localization system (AstroLoc). The integration required interfacing the uVGS-2 output with the MLP’s ROS-based software architecture using existing data pipelines. The cameras available at Astrobee’s MLP (NavCam, DockCam, SpeedCam) are monochrome cameras, which degrades uVGS-2 performance as hue (color) cannot be used as part of the blob filtering logic for noise rejection. Use of a color camera stream (SciCam) available on the guest science module (HLP) was possible via an APK that publishes SciCam images to a topic available to MLP applications. uVGS-2 provides estimates of the relative pose between the follower and a beacon-equipped leader; the leader executes predefined trajectories while the follower uses uVGS-2 measurements to maintain a prescribed relative position and orientation. Leader–follower configurations introduce line-of-sight constraints, partial occlusions, and varying illumination conditions for the Follower.

1.2.2. Sensor Fusion of uVGS-2 with Astrobee's Localization Pipeline

Leader–follower maneuvers in the SVGS-1 mission revealed critical vulnerabilities associated with line-of-sight dependency and environmental interference, leading to degradation in localization performance due to occlusion of visual features, increased distance from reference surfaces, and the presence of high-intensity illumination from the nearby beacon, leading to loss of localization and unstable control behavior. Reliance on far-field visual landmarks for localization can be mitigated if an additional pose sensor such as SVGS is available via sensor fusion [26]. Astrobee's graph-based localizer (AstroLoc) uses a map-based factor graph framework to combine visual landmarks and inertial measurements to estimate the robot's 6-DOF pose vector [27]. By incorporating uVGS-2 measurements as additional constraints within the factor graph, the system can maintain accurate state estimates in the presence of partial occlusions or degraded illumination conditions. The implementation is based on the transfer of uVGS-2 messages to AstroLoc using the docking camera AR tag pipeline. Approach and docking in Astrobee is based on AR tags located near Astrobee's docking ports on ISS. The uVGS-2 integration strategy required modification of the existing localization pipeline, replacing the output of the AR tag detection module with the uVGS-2 measurement stream. The substitution is facilitated by the fact that both uVGS-2 and the AR tag detection module output a 6-DOF pose estimate relative to the camera frame, effectively transforming the AR pipeline into a uVGS-based localization source while preserving the underlying localization architecture.

A second approach incorporates uVGS-2 measurements as updates within an EKF framework, combined with inertial propagation from IMU data [28]. This enables continuous state estimation even in the absence of valid uVGS-2 measurements, leveraging inertial data to propagate the state between updates. Analytical IMU propagation, based on frameworks such as GTSAM and VINS, provides a computationally efficient means of integrating inertial measurements. A third approach involved incorporating uVGS-2 measurements within the AstroLoc factor graph architecture, enabling simultaneous optimization of all available measurements [27]. This provides the highest level of robustness and accuracy, as it accounts for correlations between different data sources and enables global consistency of the state estimate. The final choice of sensor fusion implementation (AR pipeline) reflected a balance between computational efficiency, integration complexity, and performance. Fusion of uVGS-2 with AstroLoc enables reliable localization even under challenging conditions such as intermittent loss of beacon sight, illumination disturbances, and limited feature availability. Ground testing at NASA Ames' Granite Lab demonstrated that the fused system sustains accurate localization in scenarios where standalone methods fail, in particular when the follower's field of view is blocked by the leader. The fusion framework can be implemented with the available cameras, including a color image stream (from the SciCam on HLP) to improve blob detection based on hue.

1.2.3. Tetrahedral Beacons

3D beacons enable 6-DOF photogrammetric pose estimation by using four non-coplanar fiducial points arranged to provide full-rank observability in 3D space. Non-coplanarity eliminates degeneracies associated with planar target geometries (e.g., AR tags or Apriltags). The vector set defined by inter-point displacements is linearly independent, which provides a well-conditioned solution space for the perspective-n-point formulation [29]. This enables direct recovery of camera pose from a single image observation without reliance on temporal integration or multi-view reconstruction.

A key advantage of tetrahedral beacons lies in their resistance to geometric ambiguity under projection. Planar fiducial systems (tags) can be ill-conditioned when observed under oblique viewing angles, where perspective distortion can collapse spatial information and produce non-unique pose solutions. In contrast, the volumetric distribution of tetrahedral targets preserves depth diversity across all viewing directions, maintaining strong discriminability in image-space projections. This significantly improves numerical conditioning in pose estimation and reduces sensitivity to measurement noise during optimization. The configuration also mitigates singularities associated with collinearity or coplanarity in projected feature sets, ensuring that pose recovery remains stable even under partial distortion or suboptimal imaging conditions. The beacon includes actively illuminated or retroreflective fiducial elements whose spatial coordinates are precisely defined. Active illumination enhances detection reliability in low-light environments by producing high-contrast features; correspondence between blobs and beacon points is resolved using geometric checks based on inter-point distances and relative angles.

Pose estimation based on tetrahedral configurations can degrade under extreme viewing angles, when projected features approach near-collinear configurations that reduce observability. Additional ambiguities may arise from symmetric or near-symmetric feature distributions that complicate correspondence assignment. uVGS-2 incorporates geometric validation that enforces consistency constraints on inter-point relationships, rejecting physically inconsistent detections prior to optimization: only geometrically valid feature sets contribute to state estimation.

2. Materials and Methods

The uVGS-2 framework was comprehensively redeveloped relative to the original SVGS. An enhanced multi-stage image preprocessing pipeline is used, along with a novel formulation for solving the perspective-4-point (P4P) problem [30], based on a Lie algebra linearization of the optimization Jacobian [24,25] and the Google Ceres library [8]. The preprocessing pipeline starts by transforming the raw image $I(x, y)$, where x and y denote discrete pixel coordinates in the image plane domain $\Omega \subset \mathbb{Z}^2$, into a normalized and noise-suppressed representation $I_n(x, y) \in [0, 1]$, maximizing feature separation in environments such as the International Space Station (ISS), where non-uniform lighting, reflections, and illumination noise degrade raw image quality. The pose estimation problem is formulated as a nonlinear least-squares optimization over the special Euclidean group $SE(3)$, where the rigid-body transformation is defined as $\mathbf{T} = (\mathbf{R}, \mathbf{t})$, with $\mathbf{R} \in SO(3)$ representing the rotation matrix satisfying $\mathbf{R}^T\mathbf{R} = \mathbf{I}$ and $\det(\mathbf{R}) = 1$, and $\mathbf{t} \in \mathbb{R}^3$ representing the translation vector. For a given set of 3D fiducial points $\mathbf{P}_i = \left(X_i, Y_i, Z_i\right)^T$ expressed in the target frame, their projection into the image plane is defined as follows:

$$\mathbf{p}_i = \pi(\mathbf{K}(\mathbf{R}\mathbf{P}_i + \mathbf{t})) \quad (1)$$

where $\mathbf{K}$ is the intrinsic camera matrix and $\pi(\cdot)$ denotes the perspective division mapping homogeneous coordinates to pixel coordinates. The optimization minimizes the reprojection error defined as $\mathbf{e}_i = \mathbf{p}_i^{obs} - \mathbf{p}_i^{proj}$, where $\mathbf{p}_i^{obs}$ is the observed centroid of blob i, and $\mathbf{p}_i^{proj}$ is the predicted projection. Unlike standard implementations that rely on numerical differentiation to estimate the Jacobian of the error surface, uVGS-2 derives the Jacobian $\mathbf{J} = \frac{\partial \mathbf{e}}{\partial \boldsymbol{\xi}}$ analytically using Lie algebra in $\mathfrak{se}(3)$, where $\boldsymbol{\xi} \in \mathbb{R}^6$ is the representation of the pose increment in the tangent space, composed of three rotational and three translational components. This formulation enables linearization of the nonlinear optimization problem as $\mathbf{e}(\boldsymbol{\xi}) \approx \mathbf{e}(0) + \mathbf{J}\boldsymbol{\xi}$, significantly improving convergence rate, numerical stability, and computational efficiency, which are essential for real-time deployment in resource-constrained platforms.

The uVGS-2 elements are presented as follows: image pre-processing (Section 2.1), blob detection and sorting (Sections 2.2 and 2.4), the uVGS-2 engine (Section 2.5), the native factor-graph pose-graph localization system (Section 2.6), uVGS-2 sensor fusion with Astroloc (Section 2.7) and ROS implementation considerations (Section 2.8).

2.1. Image Preprocessing

Image preprocessing in uVGS-2 is a sequence of transformations applied to the raw image $I(x, y)$, which generates a binary segmentation $B(x, y)$ that in turn improves the detectability of fiducial markers while suppressing noise and irrelevant background structures. The process begins with spatial decimation, defined as $I_d(x, y) = I(\alpha x, \alpha y)$, where $\alpha \in [0, 1]$ is a scaling factor reducing resolution from $W \times H$ to $\alpha W \times \alpha H$, decreasing computational load while preserving salient structures. Intensity normalization is then applied, using min–max scaling $I_n(x, y) = \frac{I_d(x,y) - I_{min}}{I_{max} - I_{min}}$, where I_{min} and I_{max} are the minimum and maximum pixel intensities in the decimated image, respectively, ensuring that the dynamic range is mapped to $[0, 1]$. Normalization stabilizes subsequent thresholding under varying illumination conditions. A Gaussian filter is then applied, defined as $I_g(x, y) = (I_n * G)(x, y)$, where $*$ denotes convolution and $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ is the Gaussian kernel with standard deviation σ, which controls the level of smoothing. This operation attenuates high-frequency noise components while preserving low-frequency structures (fiducial markers).

Adaptive thresholding is then applied to generate a binary image $B(x, y)$, defined as $B(x, y) = 1$ if $I_g(x, y) > T(x, y)$, and $B(x, y) = 0$ otherwise, where the threshold $T(x, y)$ is computed locally as $T(x, y) = \mu_w(x, y) + k \cdot \sigma_w(x, y)$, with μ_w and σ_w the respective mean and standard deviation of intensities within a local window w, and k a tuning parameter. This enables robust segmentation under varying illumination conditions.

The binary image is further refined using morphological operations. Erosion is defined as $B_e = B \ominus S$, and dilation as $B_d = B \oplus S$, with S a structuring element. Erosion removes small isolated noise regions, while dilation restores valid features; their combination in opening $(B \ominus S) \oplus S$ and closing $(B \oplus S) \ominus S$ operations provides noise suppression with structural preservation. The output of preprocessing is a binary image $B(x, y) \in$ in which fiducial markers are well-defined. Preprocessing is implemented to be configurable, allowing parameters such as α, σ, k, and structuring element size to be tuned dynamically (via launch file) to accommodate sensor characteristics and environmental conditions (low-light environment, high dynamic range, cluttered background).

2.2. Blob Finding and Blob Selection

Blob detection in uVGS-2 must be robust to environmental variability and noise. Following generation of the binary image $\boldsymbol{B}(\boldsymbol{x}, \boldsymbol{y})$, connected components $\boldsymbol{\Omega_k} \subset \mathbb{Z}^2$ are identified as candidate blobs. Connectivity (8-neighborhood criterion) ensures that all pixels within a blob are spatially contiguous. The boundary of blob candidates is then extracted by a contour-following algorithm derived from the Suzuki–Abe method, which traces only the outer contours $\boldsymbol{\mathcal{C}_k} = \{(\boldsymbol{x_i}, \boldsymbol{y_i})\}_{\boldsymbol{i=1}}^{\boldsymbol{N_k}}$, and avoids hierarchical contour processing, which is justified as blobs are solid luminous regions without holes [12]. Contour representation supports downstream processing that requires ordered boundary information. Blob detection scales linearly with the number of pixels.

Candidate blobs are processed to identify spurious blobs. The primary descriptor is area $\boldsymbol{A_k} = \boldsymbol{m}_{00}$, computed from the zeroth-order spatial moment $\boldsymbol{m}_{00} = \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \boldsymbol{\Omega_k}} 1$, representing the number of pixels in the blob. The centroid $(\boldsymbol{x_c}, \boldsymbol{y_c})$ is computed using first-order moments $\boldsymbol{m}_{10}$ and $\boldsymbol{m}_{01}$, providing a subpixel estimate of blob location. Shape characterization uses the circularity metric $\boldsymbol{C_k} = \frac{4\pi A_k}{P_k^2}$, where $\boldsymbol{P_k}$ is the perimeter derived from the contour length: circularity ~1 is consistent with the expected projection of LED-based markers. Additional geometric descriptors for feature validation are bounding box dimensions and aspect ratio. Hue information $\boldsymbol{H_k}$ is extracted in the HSV space, enabling additional discrimination in scenarios with background clutter or reflective surfaces.

A blob $\boldsymbol{k}$ is retained if it satisfies $\boldsymbol{A_{min}} \leq \boldsymbol{A_k} \leq \boldsymbol{A_{max}}$, $\boldsymbol{C_k} \geq \boldsymbol{C_{min}}$, and $\boldsymbol{H_k} \in [\boldsymbol{H_{min}}, \boldsymbol{H_{max}}]$, where thresholds are determined based on marker geometry, camera resolution, and operational distance. Thresholds can be adapted to environmental conditions (illumination levels, noise) via parameters available in a launch file. Blob set candidates are evaluated using minimum and maximum allowable distances between candidate blobs based on the known dimensions of the tetrahedral beacon. This reduces the candidate set to a single subset $\boldsymbol{\mathcal{B}} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_4\}$ that is used for P4P pose estimation. Only high-confidence blobs sets are passed to subsequent stages.

2.3. uVGS-2 Features

uVGS-2 is implemented as a robot operating system (ROS) node in C++, facilitating modularity and integration in robotic systems by leveraging its publish–subscribe communication model to exchange data with other subsystems such as localization, guidance, and control modules. The output of uVGS-2 is the estimated beacon pose $\mathrm{T} = [\mathrm{R} \mid \mathrm{t}]$ in the camera coordinate frame. As uVGS-2 can operate on images published by any ROS camera node of compatible image format, it becomes independent from specific camera drivers, enabling compatibility with a wide range of imaging systems and hardware platforms, lens configurations and distortion models. The uVGS-2 camera model is defined through an intrinsic matrix K, that incorporates focal lengths and principal point offsets, while distortion is modeled using parameterized functions that are calibrated offline. uVGS-2 supports both single-channel and multi-channel image processing, enabling operation with both monochrome and color cameras, though color image streams give better performance as hue can be used to filter reflections or bright objects from blobs.

uVGS-2 is optimized for deterministic real-time performance through multi-threading and efficient resource management. Image preprocessing and blob detection are parallelized to exploit available CPU cores, reducing latency and increasing throughput. uVGS-2 can deliver high-frequency pose updates while maintaining compatibility with other onboard processes [31]. The pose estimation engine leverages state-of-the-art optimization libraries (Google Ceres) to achieve high accuracy with reduced computational overhead [8]. uVGS-2 modular architecture and compatibility with both ROS and ROS2 [32] enables integration with autonomy pipelines. The ability to operate as a standalone sensor or as part of a sensor fusion framework positions uVGS-2 as a state-of-the-art solution for vision-based navigation in spacecraft, drone, and robotic applications.

2.4. Blob Sorting in uVGS-2

Blob sorting addresses the deterministic correspondence problem between detected image-space features and their known three-dimensional counterparts defined by the tetrahedral beacon geometry. Following blob selection, the system operates on a reduced candidate set $\boldsymbol{\mathcal{B}} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_4\}$, where each blob is characterized by its centroid coordinates $\mathrm{p}_i = (\boldsymbol{x_i}, \boldsymbol{y_i})$ in the image plane. The objective of blob sorting is to establish a bijective mapping between the detected centroids and the ordered set of 3D fiducial points $\boldsymbol{\mathcal{P}} = \{\mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3, \mathrm{P}_4\}$, where $\mathrm{P}_i = (\boldsymbol{X_i}, \boldsymbol{Y_i}, \boldsymbol{Z_i})$ are expressed in the beacon reference frame. The approach is based on geometric invariants that are preserved under projective transformations, specifically pairwise Euclidean distances and relative orientation relationships in the image plane. The algorithm begins by computing the complete set of pairwise squared distances $\boldsymbol{d_{ij}} = (x_i - x_j)^2 + (y_i - y_j)^2$, forming a symmetric distance matrix that encodes the spatial distribution of the detected points. The maximum distance $\boldsymbol{d_{max}}$ corresponds to the longest projected edge of the tetrahedral configuration, and the associated point pair is designated as the primary baseline.

After baseline identification, the algorithm computes the midpoint $\mathrm{m} = \frac{1}{2}(\mathrm{p}_i + \mathrm{p}_j)$ of the baseline endpoints and evaluates the relative positions of the remaining two points with respect to this midpoint. The distances $d_{m,k} = \| \mathrm{p}_k - \mathrm{m} \|$ are computed for the remaining blobs and used to classify the points within the tetrahedral structure, exploiting the asymmetry of the tetrahedral geometry, where the projection of the apex point differs from that of the base vertices [8]. By assigning the closer point to the midpoint as one vertex and the farther point as another, the algorithm establishes a partial ordering that is invariant to global translation and rotation in the image plane. This step reduces ambiguity and ensures consistent labeling across frames, even under varying viewing conditions. To finalize ordering and eliminate potential mirror ambiguities, the algorithm employs an orientation test based on the sign of the cross product between vectors defined in the image plane. For three points $\mathrm{p}_i, \mathrm{p}_j, \mathrm{p}_k$, the scalar cross product $z = (x_j - x_i)(y_k - y_i) - \left(y_j - y_i\right)(x_k - x_i)$ is calculated, where the sign of z indicates the orientation (clockwise or counterclockwise) of the triplet [12]. Orientation tests provide a computationally efficient mechanism for enforcing geometric consistency, avoiding the need for exhaustive permutation searches or probabilistic matching techniques. The blob sorting algorithm includes validation checks to detect degenerate configurations, such as near-collinear projections or insufficient spatial separation between points.

2.5. uVGS-2 Engine for Pose Estimation

The core module for estimating the six-degrees-of-freedom pose of the target relative to the camera frame is formulated as a nonlinear optimization problem over the special Euclidean group $\mathfrak{se}(3)$. The objective is to determine the transformation $\mathrm{T} = (\mathrm{R}, \mathrm{t})$ that minimizes the reprojection error between observed image points p_i^{obs} and the projections of known 3D points P_i under the estimated pose. The projection model is defined as $\mathrm{p}_i^{proj} = \pi(\mathrm{K}(\mathrm{RP}_i + \mathrm{t}))$, where K is the intrinsic camera matrix containing focal lengths and principal point coordinates, and $\pi(\cdot)$ denotes the perspective division operation mapping homogeneous coordinates to pixel coordinates. The optimization problem is expressed as the minimization of the cost function $J = \sum_{i=1}^{4} \|\mathrm{p}_i^{obs} - \mathrm{p}_i^{proj}\|^2$, where the summation is performed over the four fiducial points of the tetrahedral beacon. This formulation ensures that the estimated pose minimizes the discrepancy between observed and predicted feature locations in a least-squares sense, providing an optimal solution under Gaussian noise assumptions [25].

A key innovation in the uVGS-2 solver lies in the use of Lie algebra in $\mathfrak{se}(3)$ [24,25] to derive an analytical Jacobian of the optimization of reprojection error, enabling efficient and numerically stable linearization of the nonlinear optimization problem. The pose increment is represented as a vector $\xi = (\omega, \mathrm{v}) \in \mathbb{R}^6$, where $\omega \in \mathbb{R}^3$ represents the rotational component and $\mathrm{v} \in \mathbb{R}^3$ the translational component of the pose estimate. The transformation update is expressed through the exponential map $\mathrm{T} \leftarrow \exp\left(\hat{\xi}\right)\mathrm{T}$, where $\hat{\xi}$ is the skew-symmetric matrix representation of ξ. The Jacobian $\mathrm{J} = \frac{\partial \mathrm{e}}{\partial \xi}$ is derived analytically by differentiating the reprojection error with respect to the pose parameters, eliminating the need for numerical finite-difference approximations. This analytical formulation significantly reduces computational overhead and improves numerical accuracy, particularly in scenarios with small perturbations or at high update rates. The linearized system is then solved iteratively using methods such as Dogleg and Dense Schur optimization, ensuring rapid convergence to the optimal solution [12]. The solver incorporates convergence criteria based on the norm of the update vector $\| \xi \|$ and the reduction in cost function ΔJ, ensuring that iterations terminate when a stable solution is reached. The solver includes validation tests to assess the quality of the estimated pose based on error residuals and geometric consistency checks.

One of the main bottlenecks in the solution of the perspective-n-point (PnP) problem lies in the evaluation of the gradient of the reprojection error surface as function of the parameters to be estimated. To eliminate the computational overhead and precision loss associated with numerical finite-difference approximations of the Jacobian, the uVGS-2 engine computes an analytical Jacobian evaluated on the tangent space manifold of the special Euclidean group $\mathfrak{se}(3)$ [24,25]. Let a fiducial beacon point in the world frame be denoted as $P_i \in \mathbb{R}^3$, which is transformed into the camera frame as $p'_i = [x'_i, y'_i, z'_i]^T = RP_i + t$, where $T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in \mathfrak{se}(3)$. Under a minimal Lie algebra perturbation defined by the twist coordinates $\xi = [\omega, v]^T \in \mathbb{R}^6$ (comprising 3 rotational and 3 translational components), the transformation update maps via the exponential map $\mathrm{T} \leftarrow \exp\left(\hat{\xi}\right)\mathrm{T}$. The analytical expression for the 2×6 reprojection error Jacobian matrix J_i for each observed fiducial point P_i is evaluated on the tangent space of the special Euclidean group $\mathfrak{se}(3)$ by the following chain rule formulation:

$$J_i = \frac{\partial e_i}{\partial \xi} = \frac{\partial e_i}{\partial p'_i} \frac{\partial p'_i}{\partial \xi} \tag{2}$$

where $\frac{\partial e_i}{\partial p'_i}$ represents the 2×3 derivative matrix of the camera perspective projection given by the horizontal and vertical focal lengths $\left(f_x, f_y\right)$ and the transformed camera-frame coordinates $p'_i = [x'_i, y'_i, z'_i]^T$. The analytical differentiation of the camera projection model with respect to the incremental pose perturbation is derived using the chain rule on the $\mathfrak{se}(3)$ manifold [31]. This yields the following block-structured Jacobian matrix:

$$\frac{\partial e_i}{\partial p'_i} = -\begin{bmatrix} \frac{f_x}{z'_i} & 0 & -\frac{f_x x'_i}{(z'_i)^2} \\ 0 & \frac{f_y}{z'_i} & -\frac{f_y y'_i}{(z'_i)^2} \end{bmatrix} \tag{3}$$

where $\frac{\partial p'_i}{\partial \xi}$ is the 3×6 kinematic derivative matrix mapping the incremental Lie algebra twist perturbation $\xi = [\omega, v]^T \in \mathbb{R}^6$, which expands analytically using the skew-symmetric cross-product matrix operator $[p'_i]^\wedge$. To avoid singularities and maintain the orthogonality constraints of the rotation matrix during the Levenberg–Marquardt optimization, a local perturbation vector $\xi \in \mathfrak{se}(3)$ is mapped to the Lie group via the following exponential map [25]:

$$\frac{\partial p'_i}{\partial \xi} = \begin{bmatrix} I_{3\times 3} & -[p'_i]^\wedge \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & z'_i & -y'_i \\ 0 & 1 & 0 & -z'_i & 0 & x'_i \\ 0 & 0 & 1 & y'_i & -x'_i & 0 \end{bmatrix} \tag{4}$$

The dimension of the stacked Jacobian J for N visible landmarks is $2N \times 6$. By directly calculating the product of these matrix blocks, the closed-form analytical Jacobian bypasses iterative residual evaluations. This formulation guarantees deterministic execution time and scales the system's real-time performance on processing nodes.

The optimization vector $\Delta\xi \in \mathbb{R}^6$ represents the minimal local twist perturbation coordinates mapped via the Lie algebra $\mathfrak{se}(3)$ to update the current transformation matrix on the manifold as $\mathrm{T} \leftarrow \exp(\xi^\wedge)\mathrm{T}$. In this parameterization, the Levenberg–Marquardt solver iteratively computes the update step by solving the regularized normal equations $\left(J^T J + \mu I\right)\Delta\xi = -J^T e$, where J denotes the $2N \times 6$ analytical Jacobian matrix stack, e represents the image-plane reprojection residual error vector, and μ is the dynamic damping factor used to regulate convergence. State estimation within the unconstrained tangent space isolates the 6-DOF estimation from the geometric distortions and singularities typical of conventional Euler-angle parametrizations.

2.6. NASA's Astrobee Localization System (AstroLoc)

AstroLoc [27] is a visual–inertial navigation system (VINS) designed for Astrobee's navigation aboard the International Space Station under computational, perceptual, and dynamic constraints. Unlike terrestrial VINS implementations, AstroLoc must operate in a microgravity environment where the absence of sustained gravitational acceleration alters inertial observability. The system is also constrained by onboard processing limitations, requiring a localization architecture that prioritizes deterministic execution, numerical efficiency, and robustness to intermittent visual information. To meet these requirements, AstroLoc is implemented as a graph-based optimization system built on the GTSAM framework, where robot state estimation is formulated as a factor graph. In this representation, discrete robot poses are encoded as nodes, while sensor-derived constraints form edges connecting sequential and non-sequential states. Each state includes a position $\mathbf{t}_k \in \mathbb{R}^3$, orientation $\mathbf{R}_k \in SO(3)$, velocity $\mathbf{v}_k \in \mathbb{R}^3$, and IMU bias terms, allowing tightly coupled estimation of kinematics and sensor drift. Visual observations from environmental features and pre-integrated inertial measurements jointly constrain the optimization problem, enabling consistent trajectory estimation over time [33]. A core architectural advantage of AstroLoc is its feature map-based design, which stores previously observed landmarks and re-associates them across time, improving long-term consistency and reducing drift accumulation [34]. This is particularly relevant in the ISS environment, where repetitive geometry and limited natural texture challenge conventional feature tracking and loop-closure strategies.

Astrobee's localization [35] operates in specialized modes, including a docking configuration mode that relies on AR fiducial markers for high-precision pose estimation during docking approach maneuvers. In this mode, AR markers are detected and their corner features are directly incorporated into the factor graph as high-confidence geometric constraints, enabling accurate relative localization with respect to docking infrastructure. However, this docking-specific pipeline is functionally isolated from the standard navigation mode, which relies primarily on feature-based tracking using features from accelerated segment test (FAST) corner detection for general motion estimation. The system switches between these pipelines depending on operational context, which limits the integration of heterogeneous sensing modalities within a unified estimation framework. This separation restricts the ability to incorporate additional sources into the global optimization process. The integration of uVGS-2 addresses this limitation by using the docking AstroLoc pipeline to accept target-derived pose measurements as additional factor graph constraints [35]. This is achieved by replicating the AR-tag processing pathway and replacing its output stream with uVGS-2 pose outputs, transforming the external vision-based estimator into a native constraint source within the graph structure [36]. This modification enables simultaneous fusion of environmental features and uVGS-2 measurements, supporting both static landmark tracking and dynamic target-based navigation within a unified optimization framework. The extended system also supports multiple camera modalities, including both the MLP monochrome navigation cameras and the HLP color camera. The incorporation of these heterogeneous sensing channels transforms AstroLoc into a multi-sensor fusion architecture while preserving its original computational efficiency and real-time performance characteristics [37,38].

2.7. uVGS-2 in Astrobee—Sensor Fusion

During the SVGS-1 mission, SVGS was used as an external measurement source, operating independently from the robot's native localization pipeline, which showed critical limitations under conditions involving line-of-sight obstruction and disturbances induced by active beacon illumination. These effects resulted in the degradation of Astrobee's native localization, which relies on environmental features, compromising control performance. uVGS-2 was designed for direct integration with the Astrobee software stack as a ROS node executing on the mid-level processor (MLP), enabling real-time exchange of state information with the graph-based localizer [39]. This required a software interface to transfer 6-DOF pose estimates from uVGS-2 into Astrobee's localization framework. The 6-DOF transformation $T_{uVGS} \in SE(3)$, estimated at a rate up to 100Hz by the uVGS-2 engine, is converted into local state coordinates to feed to the AstroLoc multi-sensor factor-graph architecture. This is equivalent to adding another node to the native pose-graph localization; the transformation matrix is formulated as a 4×4 homogeneous mapping containing the optimal 3×3 rotation matrix R and the 3×1 translation vector t, as follows:

$$T_{uVGS} = \begin{bmatrix} R & t \\ 0_{1\times3} & 1 \end{bmatrix} \tag{5}$$

The step vector $\Delta\xi \in \mathbb{R}^6$ from the tangent space manifold is mapped via the Lie exponential as $T_{uVGS} \leftarrow \exp(\xi^{\wedge})T_{uVGS}$; the direct algebraic mapping eliminates the need for coordinate re-parameterizations, preserving propagation of the error covariance across the localization pipeline and ensuring estimation stability under operational conditions that could otherwise degrade output quality. In the proposed approach, the uVGS-2 pose estimator replaces the native AR tag pipeline within the AstroLoc localization [39]. Although the integration is implemented as a data pipeline swap at the software level, the underlying AstroLoc architecture processes the relative poses as factor graph constraints, minimizing residuals on the $SE(3)$ manifold as Lie exponential updates [40,41].

The uVGS–AstroLoc integration was demonstrated with both NavCam (monochrome) and SciCam (color) image streams. uVGS-2 outputs were formatted to use the interface of the existing AR tag pipeline, allowing direct substitution of fiducial marker measurements with minor modifications to the downstream processing chain, preserving compatibility with the factor graph optimization framework underlying AstroLoc. uVGS-2 could be deployed both as a standalone pose estimator and as an integrated component within the sensor fusion scheme, with the latter providing superior robustness in scenarios involving occlusions and dynamic lighting disturbances. Sensor fusion between uVGS-2 and AstroLoc involves augmenting Astrobee's native map landmark (ML) pipeline, based on sparse visual features, with externally derived pose estimates from uVGS-2, generated from beacon tracking. uVGS-2 measurements are injected as high-confidence visual landmarks via a modified AR tag pipeline. Rather than introducing a separate localization mode, the system uses compile-time flags to modify the behavior of the AR pipeline, enabling selective use of both ML (AstroLoc) and uVGS-derived measurements [39]. A critical aspect of this fusion strategy is the ability to maintain localization continuity in scenarios where one sensing modality becomes unreliable. For instance, when the follower robot's field of view is obstructed by a leader vehicle or when strong illumination from the beacon degrades feature detection, the ML pipeline may fail to provide sufficient features for accurate state estimation. In such cases, uVGS-2 measurements provide direct relative pose information that preserves estimator observability.

2.8. Build and Test uVGS-2 in Astrobee—Implementation Considerations

Ground experiments at NASA Ames' Granite Lab allowed evaluation of the functionality and performance of uVGS-2. Modifications included configuration of camera intrinsic parameters, adaptation to Astrobee-specific image topics, and integration with the ff_msgs::VisualLandmarks message interface for publishing pose estimates on the MLP. Data logging via ROS bag files captured synchronized streams of camera data, pose estimates, and EKF outputs. Sensor fusion of AstroLoc and uVGS-2 was implemented through a modification of the AR tag pipeline, enabling direct substitution of AR tag measurements with uVGS pose estimates: the ARVisualLandmarksCallback function was modified to process uVGS-generated measurements, and the sparse mapping pipeline was extended to operate concurrently with the modified AR pipeline, enabling simultaneous use of environmental features and UVGS-2 measurements. The dual-pipeline configuration transforms the localization system into a sensor fusion framework [40,42].

3. Operational Validation and Experimental Results

A benchmark test was implemented to compare UVGS-2 pose estimation accuracy in both translation and rotation with the corresponding accuracy of state-of-the-art P4P/P3P pose solvers. The test was based on synthetically generated poses with added gaussian noise. The solvers used for comparison (Fa-P4P, EPnP, and Lambda Twist) are available as open source; results of the benchmark tests are shown in Table 1.

Table 1. Comparison of percent error in translation and rotation between UVGS-2 and state-of-the-art PNP solvers, using synthetically generated poses with Gaussian noise.

% Error	UVGS-2	Fa-P4P	EPnP	Lambda Twist
Translation (% error)	0.088	1.010	0.548	0.431
σ (% error)	2.220	2.015	1.650	1.553
Rotation (% error)	0.496	1.750	1.787	1.350
σ (% error)	2.097	2.721	2.832	2.227

FaP4P is an implementation of the uncalibrated P4P algorithm of Yang Guo [30]. The algorithm solves for pose, unknown focal length, and aspect ratio. EPnP [43] is a non-iterative solution to the PnP problem of the pose estimation of a calibrated camera from n 3D-to-2D point correspondences, whose computational complexity grows linearly with n. Lambda Twist [44] is a highly accurate, fast, and robust algorithm to solve the perspective-3-point (P3P) problem to determine the precise position and orientation of a camera relative to a 3D scene using three known points. Table 1 shows significantly lower percent error in both translation and rotation for UVGS-2, compared to the other state-of-the-art methods, when using synthetic poses with added noise as described above.

Note (for results in Sections 3.1–3.4): Due to NASA's export control and controlled unclassified information (CUI) management policies regarding operational space assets, raw quantitative performance metrics, precise timing benchmarks, and specific flight data cannot be publicly disseminated. The results presented herein reflect the maximum permissible level of reporting cleared up for open access.

3.1. NASA ARC's Granite Lab—Ground Testing Considerations and In-Orbit Data Link

Ground testing at NASA Ames Research Center's Granite Lab intends to replicate implementation constraints associated with proximity operations on ISS. The Granite Lab provides a low-friction air-bearing floor that enables motion in three degrees of freedom (X, Y, and yaw), preserving the geometric relationships necessary for 6-DOF pose estimation by vision-based sensing. Granite Lab testing reproduces implementation constraints related to actual deployment on Astrobee. The mid-level processor (MLP), which hosts the uVGS-2 ROS node, imposes computational limitations in CPU utilization, memory bandwidth, and scheduling. The absence of color imaging capabilities in the MLP (monochrome NavCam and DockCam streams) introduced challenges in blob detection and false-positive rejection, as hue-based filtering could not be used. Other issues to be considered in uVGS-2 deployment on Astrobee are camera intrinsic calibration, field-of-view distortion modeling, and coordinate frame consistency between uVGS-2 outputs and the factor graph representation.

The development of motion control strategies based on uVGS-2 prompted the implementation of a hardware-in-the-loop (HIL) simulation framework. A 3-DOF gantry was built to emulate beacon motion (Figure 20), enabling validation of perception–action coupling via NASA's Rviz and Gazebo Astrobee simulation framework, enabling testing of leader–follower maneuvers and debugging of synchronization and computational constraints within the MLP. HIL simulation of perception and control is shown in Figure 21 (HIL simulation of the follower's motion node). The follower code sends motion commands created based on 6-DOF pose updates published by the uVGS-2 node. In the HIL simulation, real-time images from an Intel D455 camera is used to detect beacon features and compute the beacon's 6-DOF pose estimates, which are then used to generate motion commands within NASA's Astrobee Gazebo simulation environment.

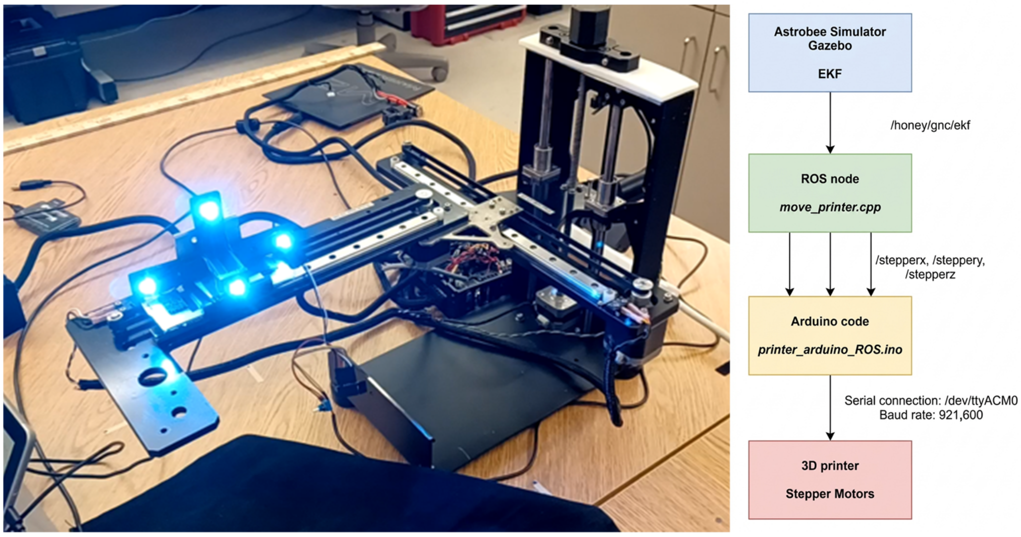

Figure 20. 3-DOF gantry for Astrobee HIL simulation. Stepper motors for each axis are controlled by a ROS node on an external Arduino board, connected to the simulation computer via USB.

To assess functionality and performance of the uVGS-2 ROS node, a hardware-in-the-loop simulation was implemented using the Inforce 6640SBC board—the same CPU as Astrobee's MLP. The implementation achieves a pose estimation update rate of 100 Hz, compared with the 10 Hz rate achieved in the Java for Android baseline implementation (SVGS) in the same CPU. uVGS-2 low-latency enabled the successful demonstration of follower maneuvers using an external camera to mimic a leader's motion (Figure 21), where Astrobee's gazebo simulation environment was used to demonstrate that the system could track beacon motion and detect all four beacon blobs during testing. The ground experiments (GT1 to GT4), conducted on Astrobee at NASA Ames Granite Lab, demonstrate the deployment of uVGS-2 as a C++ ROS node on Astrobee's mid-level processor (MLP), as well the sensor fusion integration of uVGS-2 with Astrobee's native localization framework (Astroloc).

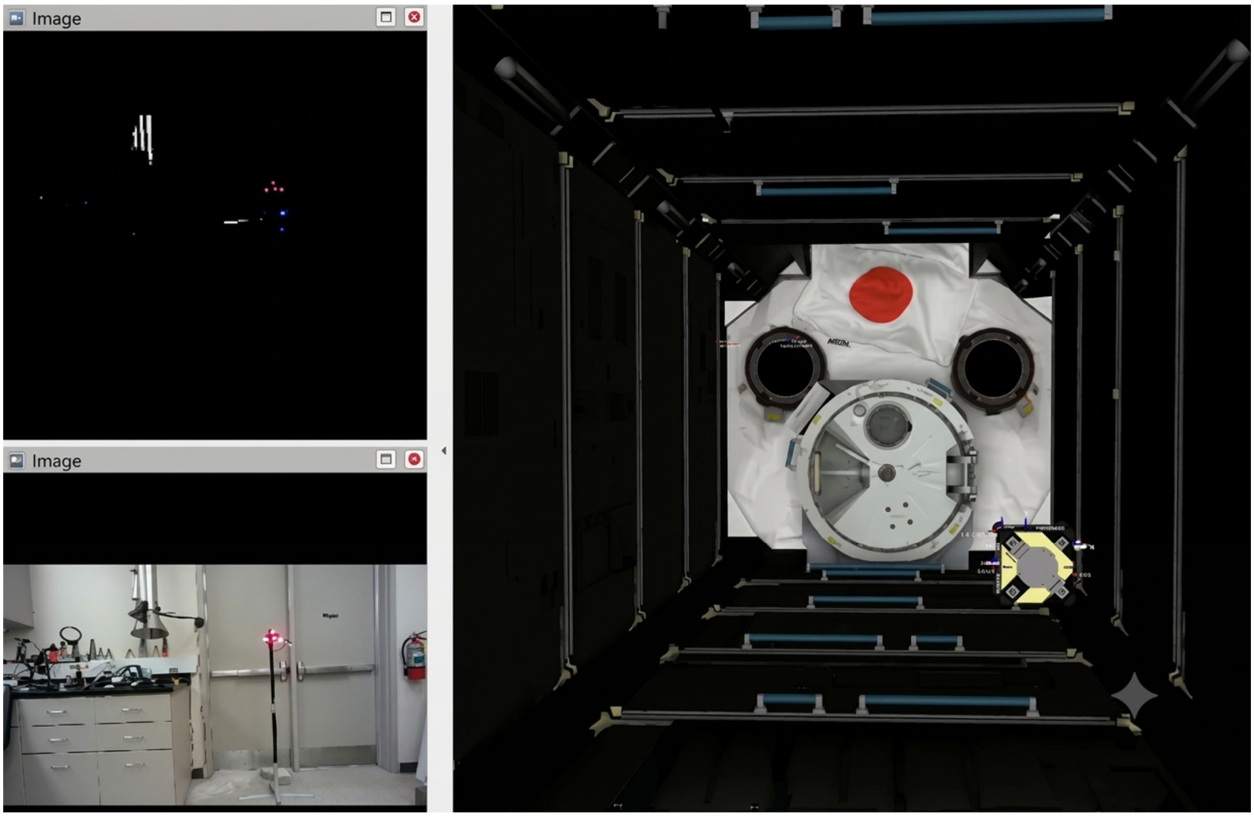

Figure 21. Hardware in the loop (HIL) simulation of the follower motion code. The follower Astrobee generates motion commands based on uVGS 6-DOF pose vectors. The HIL demonstration uses an Intel D455 camera to locate the SVGS beacon (bottom, **left**) and uses the detected blobs (top, **left**) to calculate UVGS-2n 6-DOF output. The demonstration is based on moving the camera on a square trajectory in front of the beacon, which creates motion commands in NASA's Gazebo simulation environment, demonstrating follower behavior on Astrobee.

3.2. Experiment GT-1: Demonstration of uVGS-2 Deployment on Astrobee

Ground test experiment GT-1 demonstrated the deployment and functionality of uVGS-2 in Astrobee, building and deploying uVGS-2 as a ROS C++ node to run on Astrobee's MLP. Modifications were needed to accommodate Astrobee-specific camera interfaces, including intrinsic parameters, adaptation to ROS image topics, and integration with the ff_msgs::VisualLandmarks message structure for publishing 6-DOF pose estimates. uVGS-2 execution was controlled from the SSH command line. Correct operation was verified by monitoring pose messages and checking consistency between estimated beacon positions and known robot–beacon configurations.

uVGS-2 deployment in Astrobee was demonstrated with both NavCam and DockCam, although performance degradation (failure to detect all four blobs during tests) occurred due to the inability of monochrome imaging to use hue-based blob filtering. The GT1 experiment established feasibility of deploying uVGS-2 as a ROS node on Astrobee, demonstrating its capability to operate in real time along with other MLP processes, and produce accurate 6-DOF pose estimates suitable for downstream integration with localization and control subsystems.

Figure 22 outlines Ground Test 1 (GT-1), showing the experiment layout and image processing results, including a raw NavCam image and post-processed visualization in RViz. The red markers overlaid on the processed image indicate successful blob detection and centroid extraction, which enables accurate estimation of the 6-DOF pose vector. Camera exposure, gain and uVGS-2 preprocessing parameters are editable in the corresponding launch file.

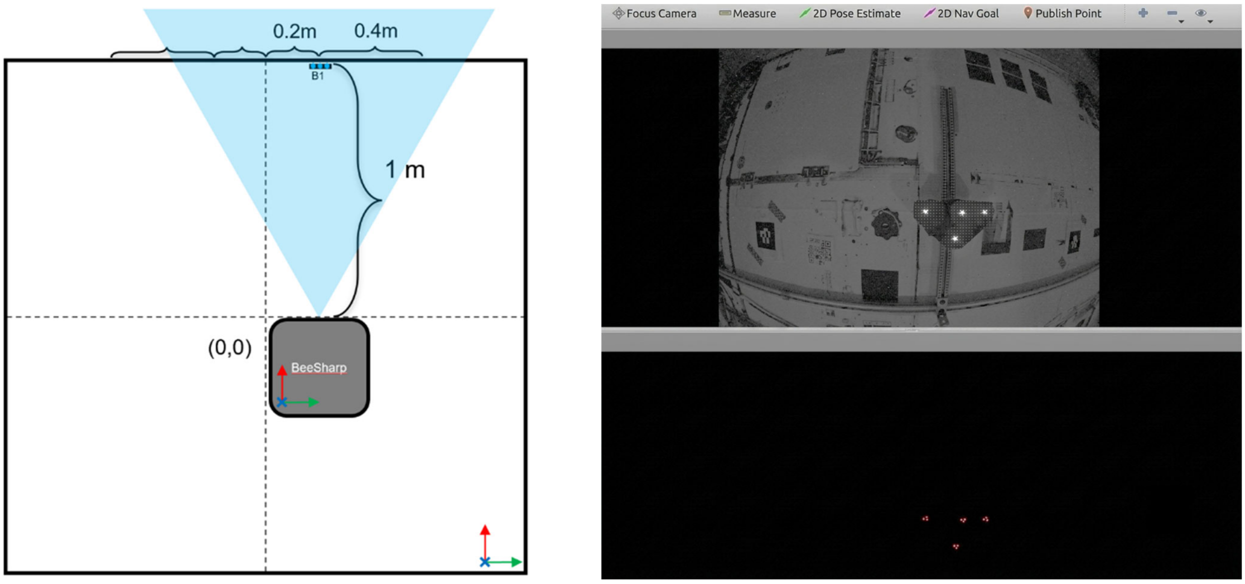

Figure 22. GT-1 experiment at NASA Ames' Granite Lab. (**Left**): Layout of the experiment, (**Right**, top): NavCam image sample, (**Right**, bottom): Post-processed image on RVIZ showing correct identification of the four beacon blobs as red dot overlays.

3.3. Ground Experiment GT-2. Demonstration of Sensor Fusion: uVGS-2 as a Data Stream That Replaces AR Tag Data in the AR Pipeline in Astrobee

Experiment GT-2 runs uVGS-2 in Astrobee as a data stream that replaces the AR tag data stream in the AR pipeline, demonstrating the feasibility of substituting AR tag inputs with uVGS-2 measurements. In the modified AR pipeline, the uVGS-2 node publishes pose data to the same interface used by the marker_tracking package, enabling seamless integration of uVGS-2 into the existing factor graph framework. A secondary objective was to determine whether the fusion of UVGS-2 data could maintain localization continuity in an occlusion scenario, where the feature-based AstroLoc would fail. Ground experiments showed that Astrobee could maintain good localization with uVGS–AstroLoc fusion even when the robot's view of the walls was significantly blocked, or when the beacon's proximity caused sensor saturation.

The attitude estimation of Euler angles by SVGS/uVGS clearly outperforms the native ML localization when detection of local landmark features degrades, as clearly shown in Figures 11 and 13, where Euler angle estimation by ML grows out of bounds. The integration of uVGS-2 with the ML pipeline also mitigates trajectory drift typically associated with vision-based localization, providing more stable and robust 6-DOF pose estimation during close-range formation flight maneuvers.

GT-2 demonstrated that uVGS-2 can replace the AR tag data stream within the AR pipeline while preserving or enhancing the robustness of the localization system. Astrobee successfully processed simultaneous inputs from both the map landmark (ML) and UVGS-modified AR pipelines, demonstrating the viability of a dual-input factor graph architecture. Localization performance remained stable even under partial or complete occlusion, as uVGS-2 measurements provided sufficient geometric constraints to maintain estimator observability. GT-2 also revealed the need for careful tuning of factor graph weights to mitigate the impact of false readings from the uVGS-2 pipeline, which could introduce localization errors if not filtered. The absence of color information (monochrome cameras on MLP were used) limits the system's ability to properly reject spurious reflections, highlighting the importance of using a color camera for optimal performance.

GT-2 demonstrated full sensor fusion of uVGS-2 with Astrobee's graph-based localizer through a modified AR pipeline, enabling simultaneous utilization of beacon-based and feature-based measurements within the AstroLoc estimation framework. The experimental setup involved commanding Astrobee to execute a square maneuver through a sequence of motion commands issued via the ground data system (GDS), while recording localization outputs under two configurations: (i) a native map landmark (ML) pipeline only and (ii) a fused ML + uVGS-2 pipeline, where the uVGS-2 node continuously published pose estimates derived from beacon observations, which were injected into the factor graph as visual landmarks via the modified AR interface.

An outline of the test setup at Ames's Granite Lab is shown in Figure 23 (left). Localization results from the robot (XY coordinates) after commanding square maneuvers as a sequence of four move commands from the GDS are shown in Figure 23 (right). Motion performance when using the native map landmark pipeline (red trace) can be compared with motion performance when using the fused uVGS-2 + map landmark solution (blue trace), showing improved motion control performance achieved through sensor fusion. The experiment also highlights the limitations of using monochrome imaging, with reduced blob discrimination leading to increased susceptibility to false detections; this motivated subsequent integration of color image streams from SciCam to enhance robustness through hue-based filtering, via an HLP APK that publishes SciCam images as a ROS topic that can be read at the MLP. Integration of uVGS-2 with AstroLoc via sensor fusion provides a robust and flexible localization solution for formation flight, even if the follower's view of the wall is blocked by the leader, or if the follower's camera is blinded by close proximity of an SVGS beacon.

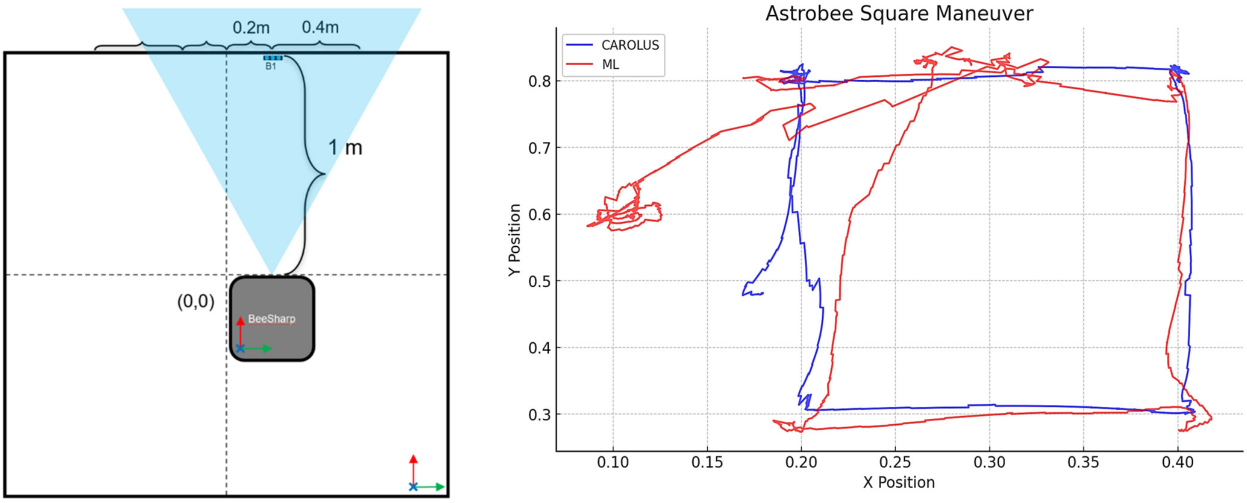

Figure 23. GT-2 experiment: Demonstration of UVGS–AstroLoc fusion via AR pipeline. (**Left**): Outline of Astrobee home position at Granite Lab for GT-2. (**Right**) Astrobee position after two repetitions of a square maneuver. The red trace shows motion performance using the native map landmark pipeline (ML), the blue trace shows motion performance using uVGS-ML sensor fusion.

3.4. Ground Experiment GT-3: Demonstration of Leader Motion Node with uVGS-Fusion

GT-3 (Figure 24, left) demonstrates a leader motion control node using fused uVGS-2 fused with AstroLoc localization to execute precise trajectory tracking. The experiment commands the leader robot to perform a square maneuver by two different methods: (i) Baseline: native AstroLoc with no sensor fusion, with manual motion commands issued through the GDS interface, and (ii) an autonomous motion script implemented as a ROS node with pose feedback from the fused localization system. A FIFO filter within the motion control node enables smoothing of pose estimates, improving command stability.

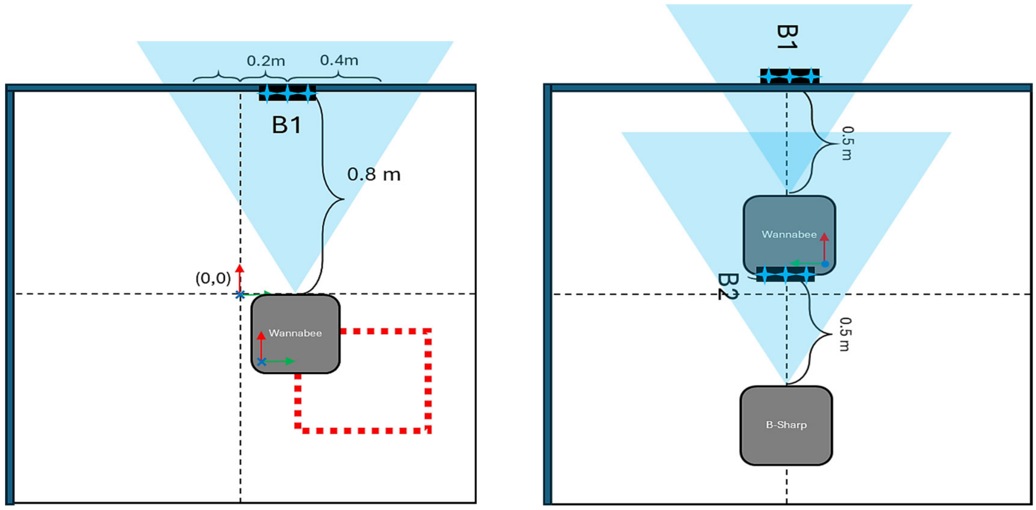

Figure 24. (**Left**): GT-3 experiment: Leader maneuver using uVGS–AstroLoc fusion and color camera (SciCam image stream). (**Right**): GT-4 experiment: Outline of leader–follower maneuver at Ames' Granite Lab. The starting home position is shown, the red trace shows the leader's trajectory.

The image input stream is from a color camera (SciCam on HLP), publishing images to a topic that can be read by uVGS-2 on the MLP. The procedure to achieve UVGS–AstroLoc sensor fusion using the SciCam image stream is as follows: (i) HLP android APK publishes SciCam-compressed images; (ii) hue for SciCam, depending on beacon color, is chosen in the camera launch file; (iii) calibration check is undertaken (extrinsics and intrinsics); (iv) uVGS pipeline is changed to compressed images and to include HUE analysis; (v) the Astrobee localizer factor graph has to consistently find world_t_dock (the transformation matrix representing the beacon's position and orientation within the ISS coordinate frame) for the follower Astrobee; (vi) factor graph options are changed for AR mode to use SciCam options; (vii) mixed graph (AR + map landmark) is enabled; and (viii) AR pipeline is changed to accept uVGS-2 outputs rather than the AR native input stream.

Results from the GT-3 experiment are shown in Figure 25. The square maneuver was repeated first using manual commands from the GDS terminal (left) and then using a leader motion script (right). A significant enhancement in trajectory accuracy was achieved when the autonomous motion node was used, as compared with manual command execution via GDS. A residual vertical tilt (~20 mm) is attributed to mechanical misalignment in the air-bearing support system that holds Astrobee. Integration of uVGS-2 with AstroLoc not only enhances localization performance but also enables motion control that outperforms waypoint manual commands in the baseline system. The extension of GT-3 to multi-robot formation is outlined in Figure 24 (right), which shows the home position for leader–follower maneuvers. The follower robot uses uVGS-2 measurements to track a beacon mounted on the leader, maintaining a prescribed relative pose throughout the maneuver.

GT-4 results are shown in Figure 26, where trajectories of both leader and follower are plotted in the XY plane. While the leader exhibits stable and accurate motion, the follower's performance deteriorates as a result of the CPU and resource limitations of the MLP when executing multiple motion commands. This underscores the importance of optimizing motion control algorithms and resource allocation in embedded GNC systems.

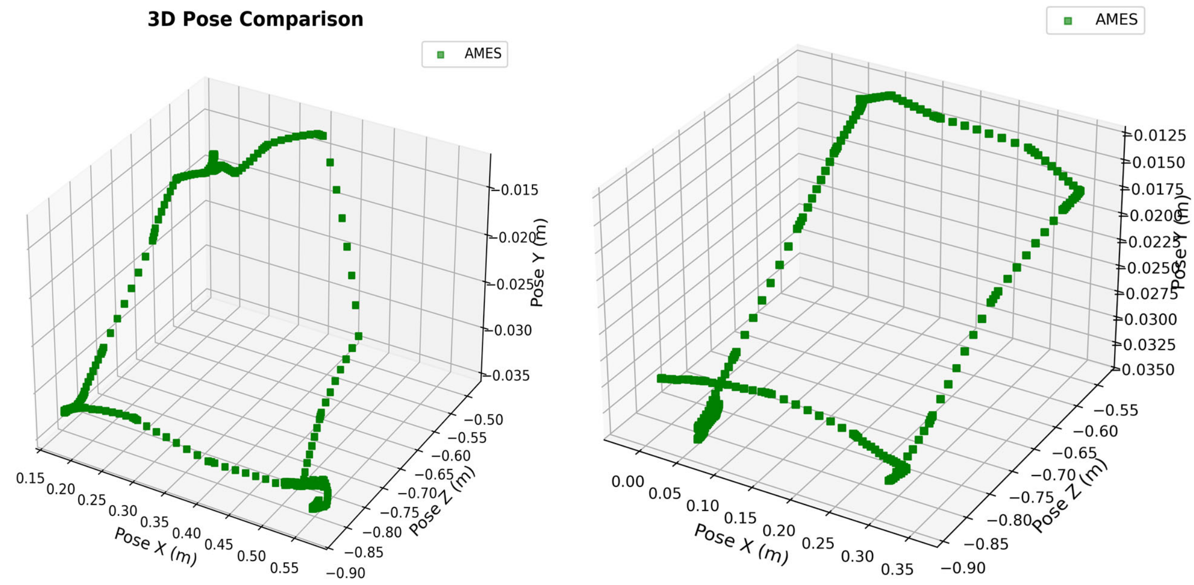

Figure 25. GT-3 experiment results. Leader maneuver with UVGS–AstroLoc fusion. (**Left**) 30 cm per side square maneuver using a sequence of manual GDS commands. (**Right**) 30 cm per side square maneuver using a leader motion script. A ~20 mm tilt in the vertical direction (in both cases) is attributed to uneven horizontal alignment of Astrobee with its air bearing support.

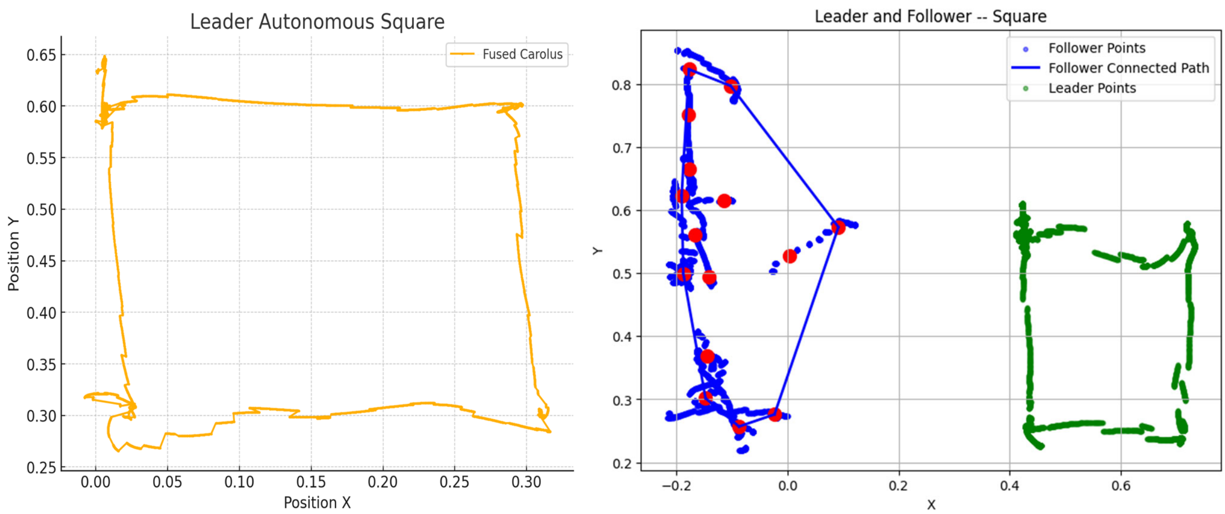

Figure 26. GT-4 experiment results: Leader–follower maneuver at Ames' Granite Lab. (**Left**) Leader motion: 30 cm per side square maneuver using uVGS–AstroLoc sensor fusion. (**Right**) Leader (green trace) and follower (blue trace) trajectories in the XY plane.

The fusion of uVGS-2 and Astroloc provides operational resilience under challenging illumination conditions (namely, proximity to a strong light source), beacon occlusions and loss of beacon line of sight, while preserving the robustness of Astrobee's native localization system. The close proximity of a beacon, which could degrade Astroloc performance, is mitigated by the availability of uVGS pose estimates in the localization pipeline, while loss of beacon sight is compensated by the default Astroloc pose estimation pipeline. The fused Astroloc–uVGS localization provides improved stability in Astrobee's motion control by reducing the probability of pose estimation failure, highlighting the robustness of the fused approach in close-range proximity maneuvers, without requiring auxiliary external updates.

4. Discussion

uVGS-2 has evolved from a proximity pose estimation sensor into a versatile vision-based navigation framework suitable for autonomous aerial and ground robotic systems operating in constrained or GPS-denied environments. Compared with the original SVGS architecture, uVGS-2 introduces substantial improvements, including ROS-native modularity, platform-independent deployment, optimized image preprocessing, and analytical optimization methods capable of real-time execution on embedded processors. Experimental validation on the Astrobee platform confirmed reliable 6-DOF pose estimation under severe illumination variations, partial occlusions, and operational conditions that degrade SVGS performance [45]. These capabilities directly address challenges encountered in UAV applications of SVGS such as precision landing [9–11], cooperative navigation, infrastructure inspection, and search-and-rescue missions [46]. Previous UAV studies demonstrated that SVGS-based navigation could maintain accurate localization in GPS-denied scenarios through fusion with visual–inertial odometry, IMU, barometer, and flight-controller states [9]. The use of illuminated tetrahedral beacons also provided greater robustness than conventional AR tag approaches, which strongly depend on favorable lighting conditions and image contrast. High-altitude landing experiments [11] validated autonomous descent from altitudes exceeding 100 m while maintaining localization continuity with intermittent beacon visibility. Unlike LiDAR or infrared beacon systems, the framework directly estimates full relative pose without requiring separate depth sensors. The successful adaptation of these concepts within uVGS-2 confirms that the sensing architecture is not limited to orbital robotics but is equally applicable to autonomous drones, cooperative robotic systems, cargo delivery, and robotic operations in visually degraded or GPS-constrained environments [47,48].

An important contribution of uVGS-2 is the incorporation of a Lie-algebra-based analytical Jacobian formulation [24,25], which significantly improves the computational efficiency of photogrammetric estimation. By eliminating numerical gradient approximations during convex optimization, processor utilization is reduced while estimation update rate increases from approximately 10 Hz in earlier SVGS implementations in Android [4] to ~100 Hz on Astrobee's mid-level processor. This improvement is critical since estimation latency directly affects guidance and control stability. Previous drone experiments have shown that SVGS measurements were often intermittent because of line-of-sight restrictions, environmental illumination, and onboard computational limitations, requiring extensive EKF tuning to maintain estimator observability. The optimized processing pipeline in uVGS-2 mitigates these limitations by enabling faster state propagation, lower computational overhead, and more stable integration with inertial and visual navigation systems. Enhanced blob extraction and filtering techniques also improve rejection of false detections caused by reflections, active light sources, and high-contrast environments, which are common in urban canyons, indoor facilities, metallic infrastructures, planetary analog environments, and spacecraft docking interfaces. In addition, the integration of uVGS-2 within Astrobee's graph-based localizer demonstrated that the system can effectively replace AR tag measurements while preserving estimator stability during beacon saturation, partial field-of-view obstruction, and intermittent target visibility [48,49]. Motion-control experiments showed improved trajectory tracking performance compared with GDS-based architectures in both simulation and robotic deployments. Although follower maneuvers revealed processor limitations associated with high-level command execution on the MLP, these results provide important insight into the computational requirements for scalable autonomous coordination [50].

5. Conclusions

uVGS-2 is a robust and computationally efficient vision-based navigation framework for autonomous proximity operations in spacecraft, drones, and mobile robot platforms. uVGS-2 advances previous SVGS implementations through a ROS-native modular architecture, enhanced image preprocessing, deterministic blob association, and a Lie-algebra-based analytical optimization implementation that significantly improves convergence speed and real-time execution performance. Experimental validation using NASA's Astrobee free-flying robot demonstrated reliable 6-DOF pose estimation under challenging operational conditions, including illumination disturbances, partial occlusions, and intermittent target visibility. The integration of uVGS-2 within Astrobee's graph-based localization framework confirmed that the system can function not only as an independent relative navigation sensor but also as a resilient complementary measurement source within multi-sensor fusion architectures. Unlike conventional localization methods that strongly depend on environmental features, LiDAR, or GNSS availability, uVGS-2 directly estimates full relative pose from monocular imagery of illuminated tetrahedral beacons, enabling accurate localization with low mass, low power consumption, and reduced computational requirements. The analytical Jacobian formulation eliminates computational overhead from numerical differentiation, increasing estimation update rate.

The robustness, portability, and computational efficiency of uVGS-2 make the framework directly applicable to autonomous UAV navigation, precision landing, and collaborative robotics. Sensor-fusion experiments confirmed that uVGS-2 can preserve localization continuity during temporary loss of visual features, beacon saturation, and line-of-sight interruptions, which are common conditions in practical applications. ROS-based implementation enables rapid deployment in robotic platforms, including VINS-based autonomy pipelines, while maintaining compatibility with other localization frameworks. The use of deterministic geometric constraints through tetrahedral beacons provides resilient relative navigation independent of environmental texture or external infrastructure. The results establish uVGS-2 as a scalable pose estimation solution capable of supporting formation flight, autonomous docking, target tracking, cooperative guidance, and precision landing in resource-constrained robotic systems.

While uVGS-2 shows significant potential to provide 6-DOF pose estimation in proximity operations, operational limitations must be acknowledged. uVGS relies on active LED illumination, and its effective operational range is thus bounded by the illumination output of the beacon, the beacon dimensions, and environmental illumination conditions. It requires the availability of a power source at the beacon location, although this can be mitigated by use of retroreflective targets, with a light source on the host robot, drone or spacecraft. Second, uVGS requires clear line of sight: occlusion of one or more fiducial points in the beacon disrupts uVGS output, requiring external multi-sensor fusion with the host's localization system (such as AstroLoc) to bridge output gaps during operation.

uVGS performance is dependent on proper tuning of preprocessing parameters such as blob dimensions, circularity, kernel size, etc. Future work must therefore include adaptive pre-processing and exposure controls to mitigate dynamic lighting transitions during orbital day/night cycles, and range-dependent adaptation of pre-processing parameters. Hardware acceleration using edge FPGA architectures is being explored to scale the uVGS framework to missions that require higher update rates, such as debris capture.

Author Contributions: Conceptualization, H.G. and I.B.; methodology, H.G.; validation, H.G. and I.B.; formal analysis, H.G. and J.C.; investigation, H.G. and J.C.; resources, H.G. and I.B.; data curation, H.G.; writing—original draft preparation, H.G. and J.C.; writing—review and editing, H.G. and J.C.; visualization, H.G.; supervision, H.G. and I.B.; project administration, H.G.; funding acquisition, H.G. and I.B. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by NASA's Marshall Space Flight Center, Cooperative Agreement 80NSSC21M0262 and 80NSSC24M0057, Dual-Use Technology Development, CAN 2021, 2023.

Data Availability Statement: The data presented in this study are available at reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest. This material is based upon work supported by the National Aeronautics and Space Administration (NASA) under Cooperative Agreement 80NSSC21M0262 and 80NSSC24M0057.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application programming interface
APK	Android package kit
AR tag	Augmented reality tag
CMOS	Complementary metal-oxide semiconductor
CPU	Central processing unit
DOF	Degrees of freedom
EKF	Extended Kalman filter
EPnP	Efficient perspective-n-point
FOV	Field of view
FPGA	Field programmable gate array
GNSS	Global navigation satellite system
IMU	Inertial measurement unit
ISS	International Space Station
LED	Light-emitting diode
ML	Map landmark
MLP	Mid-level processor
MSFC	NASA Marshall Space Flight Center
NASA	National Aeronautics and Space Administration
P4P	Perspective-4-point
ROS	Robot operating system
RPY	Roll–pitch–yaw
SVGS	Smartphone Video Guidance Sensor
UAV	Unmanned aerial vehicle
uVGS-2	Micro Video Guidance Sensor Version 2.

References

1. Hariri, N.; Gutierrez, H.; Rakoczy, J.; Howard, R.; Bertaska, I. Proximity operations and three degree-of-freedom maneuvers using the smartphone video guidance sensor. *Robotics* **2020**, *9*, 70. [CrossRef]
2. Howard, R.T.; Book, M.L.; Bryan, T.C. Video-based sensor for tracking three-dimensional targets. In *Proceedings of the Atmospheric Propagation, Adaptive Systems, and Laser Radar Technology for Remote Sensing*; SPIE: Barcelona, Spain, 2001; pp. 242–251.
3. Becker, C.; Howard, R.; Rakoczy, J. Smartphone video guidance sensor for small satellites. In Proceedings of the 27th Annual AIAA/USU Conference on Small Satellites, Logan, UT, USA, 8–13 August 2013.
4. Hariri, N.; Gutierrez, H.; Rakoczy, J.; Howard, R.; Bertaska, I. Performance characterization of the smartphone video guidance sensor as vision-based positioning system. *Sensors* **2020**, *20*, 5299. [CrossRef] [PubMed]
5. Nesnas, I.A.; Fesq, L.M.; Volpe, R.A. Autonomy for space robots: Past, present, and future. *Curr. Robot. Rep.* **2021**, *2*, 251–263. [CrossRef]
6. Janousek, J.; Marcon, P. Precision landing options in unmanned aerial vehicles. In Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW), Świnoujście, Poland, 9–12 May 2018; pp. 58–60.
7. Kalinov, I.; Safronov, E.; Agishev, R.; Kurenkov, M.; Tsetserukou, D. High-precision uav localization system for landing on a mobile collaborative robot based on an ir marker pattern recognition. In Proceedings of the 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring), Kuala Lumpur, Malaysia, 28 April–1 May 2019; pp. 1–6.
8. Silva, J. Towards Visual Inertial Navigation with Fixed Tetrahedral Targets. Ph.D. Thesis, Florida Institute of Technology, Melbourne, FL, USA, 2025.

9. Bautista, N.; Gutierrez, H.; Inness, J.; Rakoczy, J. Precision landing of a quadcopter drone by smartphone video guidance sensor in a gps-denied environment. *Sensors* **2023**, *23*, 1934. [CrossRef] [PubMed]
10. Silva, J.; Rakoczy, J.; Gutierrez, H. Precision landing comparison between smartphone video guidance sensor and IRLock by hardware-in-the-loop emulation. *CEAS Space J.* **2024**, *16*, 475–489.
11. Silva, J.; Gutierrez, H.; Bertaska, I.R.; Inness, J.P.; Rakoczy, J. High-altitude precision landing by smartphone video guidance sensor and sensor fusion. *Drones* **2024**, *8*, 37. [CrossRef]
12. Gutierrez, H. *UVGS-2: Micro Video Guidance Sensor—Revision 2*; NASA New Technology Report, e-NTR 1766866918; NASA: Washington, DC, USA, 2025.
13. Civera, J.; Davison, A.J.; Montiel, J.M. Inverse depth parametrization for monocular SLAM. *IEEE Trans. Robot.* **2008**, *24*, 932–945. [CrossRef]
14. Mizuchi, Y.; Ogura, T.; Kim, Y.; Hagiwara, Y.; Choi, Y. Vision-based markerless measurement system for relative vessel positioning. *IET Sci. Meas. Technol.* **2016**, *10*, 653–658. [CrossRef]
15. Song, M.; Ou, Z.; Castellanos, E.; Ylipiha, T.; Kämäräinen, T.; Siekkinen, M.; Ylä-Jääski, A.; Hui, P. Exploring vision-based techniques for outdoor positioning systems: A feasibility study. *IEEE Trans. Mob. Comput.* **2017**, *16*, 3361–3375. [CrossRef]
16. Ye, R.; Tao, C.; Yan, B.; Yang, T. Research on vision-based autonomous landing of unmanned aerial vehicle. In Proceedings of the 2020 IEEE 3rd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), Shenyang, China, 20–22 November 2020; pp. 348–354.
17. Zhao, B.; Chen, X.; Zhao, X.; Jiang, J.; Wei, J. Real-time UAV autonomous localization based on smartphone sensors. *Sensors* **2018**, *18*, 4161. [CrossRef] [PubMed]
18. Putra, K.T.; Wiyagi, R.O.; Mustar, M.Y. Precision landing system on H-octocopter drone using complementary filter. In Proceedings of the 2018 International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 16–17 July 2018; pp. 283–287.
19. Howard, R.T.; Bryan, T.C. DART AVGS flight results. In Proceedings of the SPIE Defense & Security Symposium, Orlando, FL, USA, 9–13 April 2007; pp. 162–171.
20. Bryan, T.C.; Howard, R.; Johnson, J.E.; Lee, J.E.; Murphy, L.; Spencer, S.H. Next generation advanced video guidance sensor. In Proceedings of the 2008 IEEE Aerospace Conference, Big Sky, MT, USA, 1–8 March 2008; pp. 1–8.
21. Howard, R.T.; Heaton, A.F.; Pinson, R.M.; Carrington, C.K. Orbital express advanced video guidance sensor. In Proceedings of the 2008 IEEE Aerospace Conference, Big Sky, MT, USA, 1–8 March 2008; pp. 1–10.
22. Bertaska, I.R.; Rakoczy, J.M.; Dankanich, J.W. *Micro Video Guidance Sensor: Versatile Relative Position Sensor for In-Space and Surface Applications NASA TechPort*; NASA: Washington, DC, USA, 2023.
23. Ahmad, J.; Warren, A. FPGA based deterministic latency image acquisition and processing system for automated driving systems. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
24. Blanco-Claraco, J.L. *A Tutorial on SE(3) Transformation Parameterizations and On-Manifold Optimization*; Universidad de Malaga: Málaga, Spain, 2022.
25. Sola, J.; Deray, J.; Atchuthan, D. A micro lie theory for state estimation in robotics. *arXiv* **2018**, arXiv:1812.01537.
26. Xin, L.; Tang, Z.; Gai, W.; Liu, H. Vision-based autonomous landing for the uav: A review. *Aerospace* **2022**, *9*, 634. [CrossRef]
27. Soussan, R.; Kumar, V.; Coltin, B.; Smith, T. Astroloc: An efficient and robust localizer for a free-flying robot. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 4106–4112.
28. Bo, C.; Li, X.-Y.; Jung, T.; Mao, X.; Tao, Y.; Yao, L. Smartloc: Push the limit of the inertial sensor based metropolitan localization using smartphone. In Proceedings of the 19th Annual International Conference on Mobile Computing & Networking, Miami, FL, USA, 30 September–4 October 2013; pp. 195–198.
29. Muratoglu, A.; Söken, H.E. Beacon number optimization for deep-space optical navigation. *Adv. Space Res.* **2026**, *77*, 12286–12301. [CrossRef]
30. Guo, Y. A novel solution to the P4P problem for an uncalibrated camera. *J. Math. Imaging Vis.* **2013**, *45*, 186–198.
31. Carlino, R.; Barlow, J.; Benavides, J.; Bualat, M.; Katterhagen, A.; Kim, Y.; Ruiz, R.G.; Smith, T.; Vargas, A.M. Astrobee free flyers: Integrated and tested. Ready for launch! In Proceedings of the International Astronautical Congress, Washington, DC, USA, 21–25 October 2019.
32. Macenski, S.; Foote, T.; Gerkey, B.; Lalancette, C.; Woodall, W. Robot Operating System 2: Design, architecture, and uses in the wild. *Sci. Robot.* **2022**, *7*, eabm6074. [CrossRef] [PubMed]
33. Tweddle, B.E.; Saenz-Otero, A. Relative computer vision-based navigation for small inspection spacecraft. *J. Guid. Control Dyn.* **2015**, *38*, 969–978. [CrossRef]
34. Barfoot, T.D. *State estimation for Robotics*; Cambridge University Press: Cambridge, UK, 2024.
35. Bualat, M.G.; Smith, T.; Smith, E.E.; Fong, T.; Wheeler, D. Astrobee: A new tool for ISS operations. In Proceedings of the 2018 SpaceOps Conference, Mareille, France, 28 May–1 June 2018; p. 2517.

36. Bualat, M.; Barlow, J.; Fong, T.; Provencher, C.; Smith, T. Astrobee: Developing a free-flying robot for the international space station. In Proceedings of the AIAA SPACE 2015 Conference and Exposition, Pasadena, CA, USA, 31 August–2 September 2015; p. 4643.
37. Chen, C.; Yang, Y.; Geneva, P.; Huang, G. FEJ2: A consistent visual-inertial state estimator design. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 9506–9512.
38. Lynen, S.; Achtelik, M.W.; Weiss, S.; Chli, M.; Siegwart, R. A robust and modular multi-sensor fusion approach applied to MAV navigation. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 3923–3929.
39. Smith, T.; Barlow, J.; Bualat, M.; Fong, T.; Provencher, C.; Sanchez, H.; Smith, E. Astrobee: A new platform for free-flying robotics on the international space station. In Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation in Space (i-SAIRAS), Beijing, China, 19–22 June 2016.
40. Smith, T.; Alexandrov, O.; Barlow, J.; Benavides, J.; Bualat, M.; Carlino, R.; Coltin, B.; Cortez, J.; Daley, E.; Feller, J. Astrobee: Free-flying robots for the international space station. *IEEE Trans. Field Robot.* **2026**, *3*, 296–337. [CrossRef]
41. Dellaert, F.; Kaess, M. Factor graphs for robot perception. *Found. Trends Robot.* **2017**, *6*, 1–139. [CrossRef]
42. Lentaris, G.; Maragos, K.; Stratakos, I.; Papadopoulos, L.; Papanikolaou, O.; Soudris, D.; Lourakis, M.; Zabulis, X.; Gonzalez-Arjona, D.; Furano, G. High-performance embedded computing in space: Evaluation of platforms for vision-based navigation. *J. Aerosp. Inf. Syst.* **2018**, *15*, 178–192. [CrossRef]
43. Lepetit, V.; Moreno-Noguer, F.; Fua, P. EPnP: An accurate O (n) solution to the P n P problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166.
44. Persson, M.; Nordberg, K. Lambda twist: An accurate fast robust perspective three point (p3p) solver. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 318–332.
45. Carlino, R.; Vargas, A.E.M.; Benavides, J.; Barlow, J.; Orosco, H.; Katterhagen, A.; Kanis, S. Lessons learned from astrobee operations on the international space station. In Proceedings of the International Space Station Research and Development Conference, Boston, MA, USA, 30 July–1 August 2024.
46. Mathurin, J.; Peter, N. Private equity investments beyond Earth orbits: Can space exploration be the new frontier for private investments? *Acta Astronaut.* **2006**, *59*, 438–444. [CrossRef]
47. Respall, V.M.; Sellami, S.; Afanasyev, I. Implementation of autonomous visual detection, tracking and landing for AR Drone 2.0 quadcopter. In Proceedings of the 2019 12th International Conference on Developments in eSystems Engineering (DeSE), Kazan, Russia, 7–10 October2019; pp. 477–482.
48. Flores-Abad, A.; Ma, O.; Pham, K.; Ulrich, S. A review of space robotics technologies for on-orbit servicing. *Prog. Aerosp. Sci.* **2014**, *68*, 1–26. [CrossRef]
49. Zhang, Y.; Li, P.; Quan, J.; Li, L.; Zhang, G.; Zhou, D. Progress, challenges, and prospects of soft robotics for space applications. *Adv. Intell. Syst.* **2023**, *5*, 2200071. [CrossRef]
50. Fluckiger, L.; Browne, K.; Coltin, B.; Fusco, J.; Morse, T.; Symington, A. Astrobee robot software: A modern software system for space. In Proceedings of the iSAIRAS (International Symposium on Artificial Intelligence, Robotics and Automation in Space), Madrid, Spain, 4–6 June 2018.