Technical Report — Hilti × Trimble × Oxford SLAM Challenge 2026

1. Summary

This report describes a solo submission to the SLAM track of the Hilti × Trimble × Oxford SLAM Challenge 2026. The system is based on OpenVINS (Hilti EUCM fork) configured for stereo visual-inertial odometry with dynamic initialization and IMU-based start-gap recovery. All 30 dataset bags were processed; 25 evaluation trajectories were submitted at 100 Hz with ≥99% temporal coverage each. The final submission (v9) achieves a final score of **2135.18 / 2500** on the challenge evaluation server. The practical contributions are: (1) enabling stereo processing for a 901× ATE improvement over the default monocular configuration, (2) dynamic initialization to handle the in-motion start of construction-site recordings, (3) backward IMU extrapolation to fill the OpenVINS initialization gap and reach the coverage threshold, and (4) a self-regulating batch pipeline that processes 30 bags in approximately 2.5 hours on a single RTX 4090. A loop closure post-processing investigation (KD-tree spatial candidate detection with a symmetric heading-cosine confidence score, DINOv2 visual confirmation, and pose-graph correction) is documented; it was ultimately not applied to v9 due to inconsistent improvement on ground-truth sequences.

2. Method

2.1 Pipeline Overview

The processing pipeline consists of three stages:

1. **Preprocessing.** Compressed ROS 2 bags (MCAP-style format with JPEG-encoded images) are decompressed into raw image topics suitable for OpenVINS. Each bag produces 15–30 GB of uncompressed data. Preprocessing is implemented in Python using OpenCV's `imdecode` and runs at approximately 0.5–1× real-time on a single CPU core.

2. **SLAM.** OpenVINS processes the raw stereo image streams (`/cam0/image_raw` , `/cam1/image_raw`) and IMU data (`/imu/data_raw`) to produce 6-DoF pose estimates at camera frame rate. The output is written to TUM format via a custom ROS 2 trajectory logger node.

3. **Post-processing.** Trajectories are extended at the start with backward IMU extrapolation to fill the OpenVINS initialization gap (§2.3), validated for temporal coverage against the scoring window, downsampled to 100 Hz, and packaged into the submission archive.

The pipeline runs on a RunPod instance (RTX 4090, 12 vCPU, 160 GB SSD) with ROS 2 Jazzy on Ubuntu 24.04.

2.2 OpenVINS Configuration

Stereo vs. Mono

The Hilti EUCM fork of OpenVINS defaults to monocular processing (`use_stereo: false`), using only one of the two calibrated fisheye cameras. In monocular VIO, metric scale is derived solely from IMU measurements — specifically from the relationship between visual parallax and accelerometer readings. When IMU bias drifts, scale estimation degrades, leading to unbounded position error.

Enabling stereo processing (`use_stereo: true`) provides direct scale observation through the known camera baseline. On the `floor_1_2025-05-05_run_1` ground-truth sequence, this single change reduced the absolute trajectory error (ATE RMSE, SE(3)-aligned) from 262 m to 0.29 m — a 901× improvement. The effect was consistent across all sequences where initialization succeeded.

Dynamic Initialization

OpenVINS's default static initialization requires the sensor platform to remain stationary while accumulating sufficient feature observations. This frequently fails in construction-site environments where:

- The operator begins moving immediately after starting the recording
- Lighting transitions (indoor/outdoor, construction lighting) cause high feature disparity in early frames
- Texture-poor surfaces (bare concrete, drywall) reduce feature count

Setting `init_dyn_use: true` enables dynamic initialization, which estimates the initial state (position, velocity, IMU biases) while the platform is in motion. This resolved initialization failures on two ground-truth sequences (UG1 and `floor_2_2025-10-28_run_1`) and improved the aggregate GT score from 330 to 427 out of 500.

Key parameters: `init_max_disparity: 50.0` (pixel threshold for feature matching during init), `init_window_time: 2.0` (seconds of data used for initialization).

CLAHE: A Negative Result

Contrast Limited Adaptive Histogram Equalization (CLAHE) was evaluated as an image preprocessing step to improve feature detection in poorly-lit areas. While CLAHE increased the number of detected features, it also increased the measured feature disparity during the initialization phase — pushing it beyond the static initialization threshold on several sequences. On the UG1 ground-truth sequence, CLAHE prevented initialization entirely (0 poses vs. 191,480 with standard histogram equalization).

This result illustrates that image-level enhancements can interact adversely with system-level assumptions. The initialization module's disparity threshold was calibrated for unenhanced images; CLAHE shifted the operating point. Dynamic initialization proved to be the correct fix — addressing the root cause (inability to initialize during motion) rather than a symptom (low feature count in dark areas).

Remaining Configuration

All other parameters were left at the Hilti fork's defaults: EUCM camera model with the provided calibration, `num_pts: 500` tracked features, `fast_threshold: 20` , and standard

histogram equalization (`histogram_method: "HISTOGRAM"`). No recalibration was performed.

2.3 IMU Backward Extrapolation

OpenVINS's initialization phase consumes the first 1.4–10.3 seconds of each recording (the exact duration depends on how quickly sufficient feature observations and motion are gathered). During this window the SLAM frontend produces no pose estimates, leaving a gap at the start of every trajectory. The challenge requires ≥99% temporal coverage of the scoring window — without intervention, several runs fall under that threshold and score 0 regardless of accuracy.

To recover this start gap, a post-processing step backward-integrates the IMU stream from the first VIO pose to the recording start. Linear acceleration and angular velocity are integrated in the body frame using the orientation reported by the first VIO sample, then chained step-by-step to fill the missing timestamps. This is dead reckoning, not SLAM: it accumulates approximately 0.5–2 m of position error at the trajectory start, depending on the gap length and motion intensity.

The trade-off is intentional. Coverage is a hard constraint (a missing prefix yields zero score); accuracy at the start is a soft constraint (the exponential scoring formula penalizes errors but does not zero out a sequence). Adding bounded dead-reckoning error to recover coverage is strictly better than scoring 0.

This step alone moved the submission from `slam_track_v6` (Score 0 due to coverage failures from a malformed extrapolation) to `slam_track_v7` (a valid coverage-passing submission), and remains active in v9.

2.4 Batch Processing Pipeline

Processing 30 bags sequentially would take approximately 15 hours. To reduce this, a parallel batch runner was developed with the following design:

Resource-aware scheduling. Each preprocessed bag occupies 15–30 GB on the local SSD. The maximum number of concurrent runs is computed as:

$$\text{max_parallel} = \min\left(\left\lfloor\frac{\text{nproc}}{12}\right\rfloor,\
\left\lfloor\frac{\text{SSD_free} - \text{buffer}}{\text{GB_per_run}}\right\rfloor\right)$$

where 12 is the approximate thread count per OpenVINS instance and `buffer` (15 GB) is reserved for ROS logs and OS overhead. This adapts automatically to different hardware configurations.

Rolling pool. Rather than processing in fixed-size waves (which idle while waiting for the slowest bag), the runner uses bash's `wait -n` to detect the completion of any child process and immediately launches the next queued bag.

Process isolation. Each concurrent run uses a unique `ROS_DOMAIN_ID` to prevent ROS 2 topic interference between parallel OpenVINS instances.

Graceful shutdown. The trajectory logger receives `SIGINT` (not `SIGKILL`) after bag playback completes, allowing rclpy's shutdown handler to flush buffered poses to disk. A grace period of 8 seconds post-playback plus 5 seconds post- `SIGINT` ensures data integrity.

With this pipeline, all 30 bags are processed in approximately 2.5 hours on a single RTX 4090 pod.

2.5 Loop Closure Investigation

Post-processing loop closure was investigated as a potential accuracy improvement on the highest-drift sequences. The investigation is documented here because it represents significant engineering effort that ultimately produced no net improvement.

Motivation

Three sequences showed catastrophic drift (RMSE > 1 m): `floor_UG2_2025-12-02_run_1` (4.04 m), `floor_7_2025-12-02_run_2` (3.88 m), and several UG1 runs. The hypothesis was that VIO drift could be corrected by detecting geometric revisit candidates and solving a pose graph.

Candidate Detection

A KD-tree spatial search identifies pose pairs (i, j) where:

- Euclidean distance < 2 m (potential revisit)
- Time gap > 60 s (not consecutive poses)

Candidates are scored with a confidence formula:

```
confidence = spatial × |cos θ| × temporal × visual_weight
  spatial  = 1 − dist / dist_threshold
  temporal = min(1.0, time_gap / 300s)
  |cos θ|  = symmetric heading score (1.0 at 0° and 180°, 0.0 at 90°)
```

The symmetric heading formula (`|cos θ|` instead of the original `(1 + cos θ)/2`) was a fix introduced during this work — the original formula zeroed out confidence for door-crossing candidates (θ ≈ 180°), which are the most geometrically meaningful revisits.

Visual Confirmation (DINOv2)

Visual similarity scores are computed using **DINOv2** (ViT-S/14) descriptors extracted from the raw fisheye images at candidate timestamps. Cosine similarity between normalized embeddings serves as a geometric-independent confirmation signal, gating the otherwise too-permissive geometric candidate set.

Type-4: Visual Start/End Loop Detection

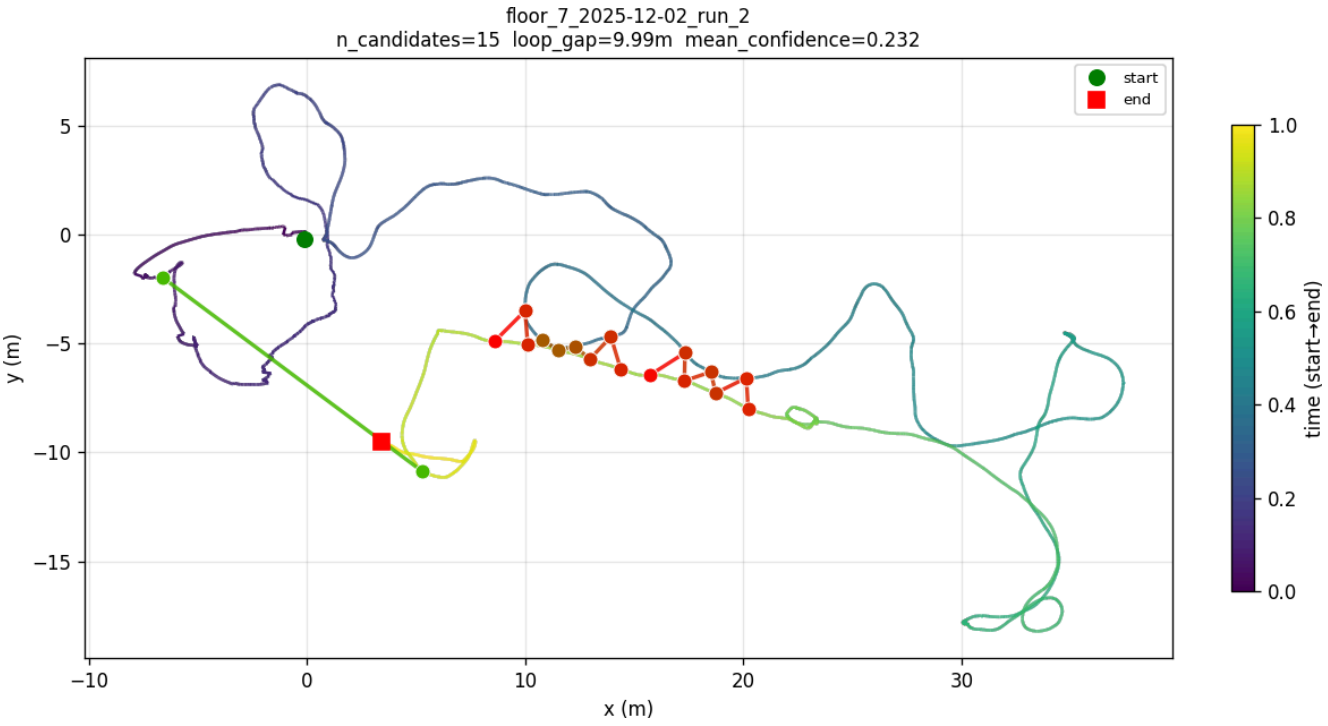
For sequences with large drift (loop gap > 2 m geometric threshold), the KD-tree cannot detect the closure. A "Type-4" candidate was added: compare a DINOv2 embedding from early in the trajectory (t_0 + 10 s) against one from the end (t_end − 10 s). If cosine similarity ≥ 0.65, a loop constraint is injected regardless of geometric distance.

Results on inspection runs:

Sequence	Type-4 similarity	Geometric gap

floor_1_2025-05-05_run_1	0.871	11.4 m
floor_2_2025-05-05_run_1	0.758	11.3 m
floor_2_2025-10-28_run_2	0.682	12.6 m
floor_7_2025-12-02_run_2	0.725	14.9 m
floor_2_2025-10-28_run_1	no match	—
UG1_2025-10-16_run_1	no match	—
floor_UG2_2025-12-02_run_1	no match	—
UG1_2025-12-02_run_1	no match	—

`floor_7_2025-12-02_run_2` was correctly flagged as a visually closed loop despite 14.9 m geometric drift — consistent with the GT trajectory showing a true closed loop.



Pose Graph Correction and Why It Was Not Applied

Detected candidates are passed to a pose graph solver (scipy `minimize` with L-BFGS-B). The objective minimizes odometry residuals (weight 10.0) and loop closure residuals (weight 1.0–2.0). The correction is applied as a smooth interpolation across the trajectory.

A grid search over five correction parameters (`dist_threshold` , `time_min` , `heading_max` , `confidence_min` , `visual_weight`) was run on the 5 GT sequences — approximately 400 parameter combinations × 5 bags ≈ 2,000 evaluations, executed across 50 parallel workers in roughly 20 minutes.

Results:

- Mean improvement: +1.1 pts on 3 of 5 GT bags
- 2 of 5 GT bags became worse
- Net effect on the 20 non-GT sequences: unobservable

Given the marginal and inconsistent improvement on the bags where GT is available, loop closure correction was **not applied** to v9. With 80% of sequences unobservable, broad application is a blind risk that the +1.1 pt average gain does not justify.

The loop closure code remains in `scripts/10_loop_close.py` for future work.

3. Results

3.1 Headline Metrics

Metric	Value
Final Score	**2135.18 / 2500**
Mean ATE (RMSE)	0.503 m
Mean Coverage	99.999%
Sequences scored > 0	25 / 25

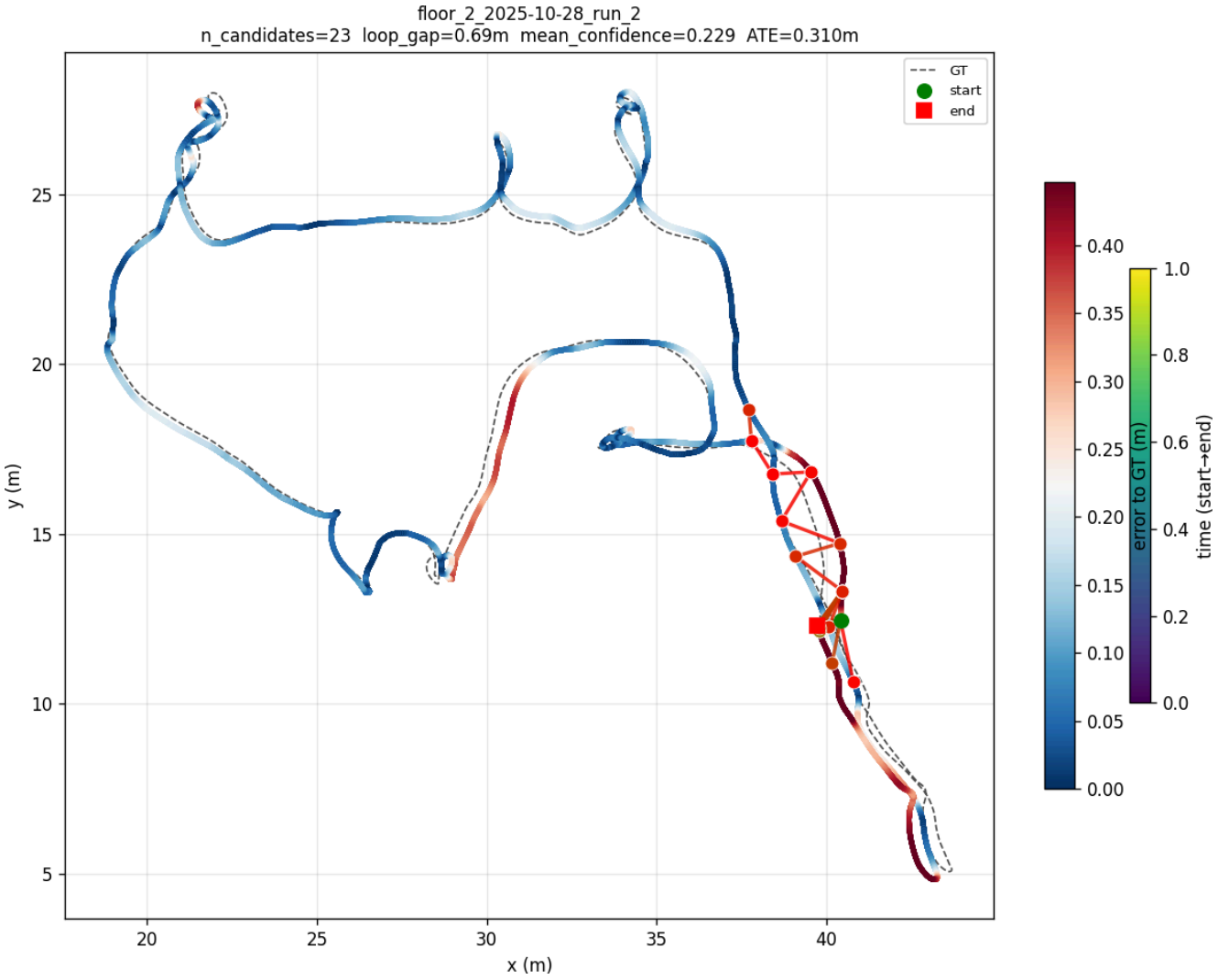
3.2 Ablation Study

The impact of each configuration change, evaluated on the ground-truth sequences:

Configuration	floor_1 ATE	floor_2_10-28_r1	UG1
Mono (default)	262 m	0 poses (init fail)	diverges
Stereo	0.29 m	0 poses (init fail)	diverges
Stereo + DynInit	0.23 m	0.20 m	0.71 m

Stereo provides the largest single improvement (scale correction). Dynamic initialization is essential for robustness — without it, two of five GT sequences fail to initialize entirely. Two additional GT sequences (`floor_2_2025-05-05_run_1` and `floor_2_2025-10-28_run_2`) showed similar behaviour under the final Stereo + DynInit configuration and are omitted here for brevity.

3.3 Representative Sequence: `floor_2_2025-10-28_run_2`



This sequence is one of the five with available ground truth. The visualization shows the pipeline's typical behavior: low error along most of the trajectory (deep blue) with localized drift on the longer segments (red/orange) that loop closure candidates target.

4. Discussion

What Worked

Stereo VIO with dynamic initialization proved to be a robust configuration for this dataset. The combination successfully processed all 30 bags — including challenging underground sequences (UG1, UG2), multi-floor traversals (`floor_7`), and sequences with significant lighting variation (`floor_EG`). No sequence-specific parameter tuning was required to achieve a final score of 2135.18 / 2500.

The batch pipeline enabled rapid iteration. Processing 30 bags in 2.5 hours meant that configuration changes could be validated across the full dataset within a single development cycle. The self-regulating parallelism adapted correctly when moving between pod configurations with different SSD capacities.

IMU backward extrapolation closed the remaining coverage gap caused by OpenVINS initialization delay, lifting all 25 evaluation runs to ≥99% coverage and unlocking a non-zero score on every sequence.

What Did Not Work

CLAHE preprocessing was counterproductive. While it improved feature detection in isolation, it disrupted the initialization module's disparity assumptions. This highlights a general risk of applying image-level enhancements without understanding their effect on downstream algorithm assumptions.

Monocular VIO produced trajectories with path lengths 16–211× longer than ground truth. While coverage was consistently ≥99% (the system tracks *that* motion occurs), metric accuracy was unusable. This confirms that monocular scale estimation from IMU alone is insufficient for construction-site conditions where IMU bias drift is significant.

Stella-VSLAM was evaluated as an alternative frontend but produced only 17 poses on the test sequence before losing tracking. The EUCM camera model support in the available build was insufficient for the dataset's extreme fisheye distortion.

Zero-velocity updates (ZUPT) had negligible effect — the camera platform was in continuous motion throughout all recordings, providing no stationary segments for ZUPT to exploit.

Loop closure post-processing was investigated thoroughly with KD-tree spatial candidate detection, DINOv2 visual confirmation (including a Type-4 visual start/end check for high-drift loops), and pose graph correction. A 2,000-evaluation grid search on the five GT sequences showed inconsistent improvement: +1.1 pts mean on 3 of 5 bags but regression on 2 of 5. Given that 80% of evaluation sequences are unobservable, broad application was rejected as too risky and the technique was not applied to v9. See §2.5.

Construction-Site Challenges

The dataset exposes several challenges specific to construction environments:

- **Lighting transitions.** Sequences crossing indoor/outdoor boundaries or passing through areas with temporary construction lighting exhibit rapid exposure changes that reduce feature matching quality.
- **Texture-poor surfaces.** Bare concrete walls, unpainted drywall, and empty floors provide few features for visual tracking. This is particularly acute in underground areas (UG1, UG2).
- **Large open spaces.** The underground parking structures have long sight lines with distant features, reducing the geometric strength of stereo matching and increasing sensitivity to depth estimation errors.
- **Scale of recordings.** Individual sequences span 5–20 minutes of continuous walking at 0.5–1.5 m/s, covering hundreds of meters. Without loop closure, drift accumulates throughout.

Limitations

The submitted system applies no loop closure. OpenVINS operates as a filter-based VIO system — once drift accumulates, it cannot be corrected by revisiting previously mapped areas. This is the primary source of remaining error, particularly on longer sequences such as `floor_UG2_2025-12-02_run_1` (4.04 m RMSE) and `floor_7_2025-12-02_run_2` (3.88 m RMSE).

Loop closure post-processing was implemented and tested (§2.5) but not applied to v9 because validation on the available GT subset did not confidently predict improvement on the unobservable majority of sequences.

Floorplan-based correction (the dataset provides DXF and PNG floorplans for most sequences) was not attempted in this submission and represents the most promising direction for drift correction in construction-site SLAM, where building floorplans are typically available.

5. Lessons Learned

1. Utilize all available sensor data. The single largest accuracy improvement came from enabling the second camera — a capability that was available but disabled by default. In practical deployments, ensuring that all sensor modalities are utilized (stereo cameras, IMU, potentially LiDAR or UWB) should be the first priority before algorithmic improvements.

2. Prefer system-level robustness over signal-level enhancement. Dynamic initialization (a system-level change) solved the initialization problem that CLAHE (a signal-level change) attempted to address but made worse. When a subsystem fails, fixing the subsystem's assumptions is more reliable than modifying its inputs.

3. Coverage and accuracy are separable concerns. Visual-inertial odometry reliably tracks continuous motion (achieving ≥99% coverage on all sequences) but cannot guarantee metric accuracy without additional constraints. These represent distinct capability tiers with different engineering requirements; meeting the coverage gate (the hard constraint) and the accuracy target (the soft constraint) require different fixes.

4. Pipeline engineering is as important as algorithm selection. Process management details — graceful shutdown for data integrity, SSD caching for I/O performance, domain isolation for parallelism, and IMU-based start-gap recovery — had direct impact on whether valid results were produced. A correct algorithm in a fragile pipeline produces no results.

5. Parameter tuning has diminishing returns once stereo and dynamic initialization are enabled. A systematic sweep over `fast_threshold` (15, 18, 20), `num_pts` (500, 650), and `max_slam` (50, 75) showed no consistent improvement over the baseline configuration; some variants produced score-zero outcomes on multiple bags. The structural drift that parameter tuning cannot address is precisely what loop closure or floorplan-based correction targets.

6. Loop closure validation needs more than five GT bags. Average-improvement metrics on a small validation subset hide per-sequence variance large enough to flip from gain to regression. With 80% of evaluation sequences unobservable, broad correction is a blind risk that average gains alone cannot justify. Selective application based on observed drift, or stronger visual gating to suppress false-positive candidates, are the natural next steps.

6. Acknowledgments

This work uses the dataset provided by Hilti, Trimble, and the Oxford Dynamic Robot Systems group as part of the 2026 SLAM Challenge. The challenge infrastructure and evaluation tools were developed by ARIAM and the Vision & Robotics Lab.

Development was assisted by Claude Code (Anthropic) for pipeline engineering, batch automation, and analysis.