drones

MDPI

*Article*

# NaviLoc: Trajectory-Level Visual Localization for GNSS-Denied UAV Navigation

**Pavel Shpagin [1,*] and Taras Panchenko [2,*]**

[1] Academia Tech, Research and Development Division, 03038 Kyiv, Ukraine
[2] Theory and Technology of Programming Department, Faculty of Computer Science and Cybernetics, Taras Shevchenko National University of Kyiv, 01033 Kyiv, Ukraine
* Correspondence: pavel.shpagin@theacademia.tech (P.S.); taras.panchenko@knu.ua (T.P.)

## Highlights

**What are the main findings?**

- NaviLoc enables training-free GNSS-denied aerial-to-satellite localization using trajectory-level fusion of image matching and motion sensing.
- On low-altitude (50–150 m) rural UAV flights, NaviLoc achieves a 19.5 m Mean Localization Error (16.0× better than prior state-of-the-art).

**What are the implications of the main finding?**

- Trajectory-level optimization handles visually repetitive rural environments, enabling reliable GPS-free navigation.
- The lightweight algorithm enables real-time CPU-only deployment (9 FPS on Raspberry Pi 5), supporting embedded autonomy.

## Abstract

Aerial-to-satellite visual localization enables GNSS-denied UAV navigation, but the appearance gap between low-altitude (50–150 m) UAV imagery and nadir satellite tiles makes per-frame Visual Place Recognition (VPR) unreliable. Under perceptual aliasing, high-similarity matches are often geographically inconsistent, so naïve anchoring fails. We introduce NaviLoc, a **training-free** three-stage trajectory-level estimator that treats VPR as a noisy measurement source and exploits Visual-Inertial Odometry (VIO) as a relative-motion prior. Stage 1 (Global Align) estimates a global SE(2) transform by maximizing an explicit trajectory-level similarity objective. Stage 2 (Refinement) performs sliding-window bounded weighted Procrustes updates. Stage 3 (Smoothing) computes a strictly convex MAP trajectory estimate that fuses VIO displacements with VPR anchors while clamping detected outliers. On a challenging low-altitude rural UAV benchmark, NaviLoc attains **19.5 m** Mean Localization Error (MLE)—a **16.0**× reduction compared to state-of-the-art localization method AnyLoc-VLAD, and **32.1**× compared to raw VIO drift. End-to-end inference runs at **9 FPS** on Raspberry Pi 5, enabling real-time embedded deployment.

**Keywords:** visual localization; UAV navigation; Visual Place Recognition; GNSS-denied navigation; satellite imagery; trajectory optimization; robust estimation

Check for updates

Academic Editor: Yuezu Lv

Received: 22 December 2025
Revised: 19 January 2026
Accepted: 26 January 2026
Published: 29 January 2026

**Copyright:** © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

## 1. Introduction

Global localization is a prerequisite for long-horizon GNSS-denied UAV autonomy. Applications such as environmental monitoring, search-and-rescue, and autonomous delivery increasingly require operation in environments where GNSS signals may be unavailable,

jammed, or spoofed. Visual-Inertial Odometry (VIO) provides locally accurate relative motion but drifts without bound, making it insufficient for missions requiring global position awareness.

A natural correction source is aerial-to-satellite Visual Place Recognition (VPR), matching onboard views to geo-referenced satellite tiles. However, as illustrated in Figure 1, the cross-view domain gap between low-altitude UAV imagery and nadir satellite maps produces frequent high-similarity false matches at geographically distant locations—even human observers find it difficult to identify correct correspondences. Per-frame VPR anchoring is therefore unreliable.

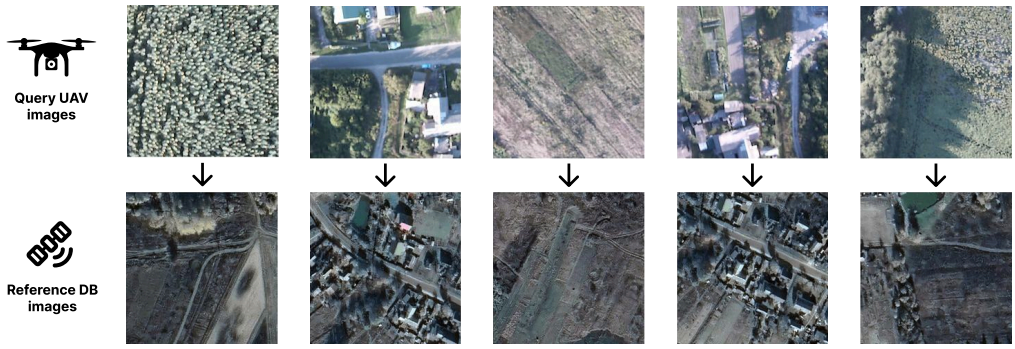



**Figure 1.** Cross-view correspondences between UAV and satellite imagery. **Upper**: UAV query frames (50–150 m AGL). **Lower**: corresponding satellite tiles matched by GPS ground truth. The visual similarity between aerial and satellite views is often ambiguous even to human observers, motivating trajectory-level optimization.

NaviLoc addresses this failure mode by estimating a *trajectory-level* solution rather than committing to individual matches. The method is training-free: it uses off-the-shelf pretrained image descriptors without domain-specific fine-tuning. NaviLoc operates in three stages: Stage 1 (Global Align) searches for a single global SE(2) transform whose implied trajectory yields consistently high *local* retrieval scores; Stage 2 (Refinement) refines the aligned trajectory through bounded weighted Procrustes updates on overlapping windows; and Stage 3 (Smoothing) computes a closed-form MAP estimate that fuses VIO displacements with VPR anchors while suppressing low-confidence anchors (see Figure 2).

Public benchmarks for this setting remain limited. We were unable to find an open-source dataset that jointly provides low-altitude (50–150 m) UAV imagery, synchronized VIO, and a geo-referenced satellite tile database suitable for trajectory-level evaluation. To study this practically relevant regime, we evaluate NaviLoc on a prepared real-world UAV-to-satellite dataset with VIO and curated satellite tiles.

*Contributions*

1. A **novel, training-free, lightweight, and embedded** three-stage trajectory-level localization method with explicit objectives and closed-form solutions.
2. An evaluation of low-altitude (50–150 m) UAV imagery showing a **19.5 m** Mean Localization Error and **16.0**× improvement over AnyLoc-VLAD [1].
3. An ablation demonstrating robustness to hyperparameter variations, and an empirical study showing that distilled ViT descriptors are more effective for *trajectory-level* alignment than larger foundation models on our benchmark.
4. An embedded implementation achieving **9 FPS** end-to-end inference on Raspberry Pi 5.

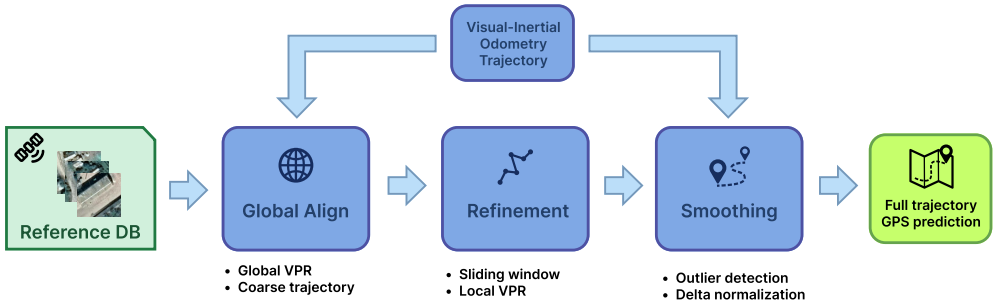



**Figure 2.** NaviLoc pipeline. **Global Align** (Stage 1): global VPR yields coarse targets used to optimize a trajectory-level similarity objective over SE(2). **Refinement** (Stage 2): windowed local VPR supports bounded weighted Procrustes updates. **Smoothing** (Stage 3): VIO deltas are enforced while low-confidence anchors are detected via similarity statistics and *clamped* to the VIO prior.

## 2. Related Work

**Visual Place Recognition.** VPR evolved from handcrafted descriptors [2] through bag-of-words [3,4] to learned representations. NetVLAD [5] introduced end-to-end learning for place recognition. AnyLoc [1] aggregates foundation model features using VLAD or GeM pooling, achieving state-of-the-art results on diverse benchmarks. MixVPR [6] proposes feature mixing for compact descriptors. These methods produce per-image descriptors without trajectory-level consistency constraints.

**Cross-Domain and Aerial Localization.** The aerial-to-satellite domain gap presents challenges distinct from ground-level VPR. CVM-Net [7] learns cross-view representations. VIGOR [8] and CVUSA [9] established benchmarks for ground-to-aerial matching. Recent UAV-specific methods [10,11] address the UAV-to-satellite gap. Our approach is complementary: we accept that individual matches are unreliable and leverage trajectory-level statistics.

**UAV-to-Satellite Geo-Localization.** Several recent works study cross-view matching between UAV imagery and satellite maps in the remote sensing literature [11–14]. These methods improve per-frame matching but still face perceptual aliasing under viewpoint and appearance changes; NaviLoc instead aggregates evidence across time using an explicit trajectory-level objective. For broader context on absolute visual localization pipelines and design choices, we refer to a recent survey in *Drones* [15]. Recent work on neural network-based recognition [16] and air safety information management [17] provides a conceptual bridge between learning-based perception methods and system-level safety considerations; our trajectory-level, training-free approach complements these by enabling robust localization without domain-specific training.

**Point Set Registration.** The Iterative Closest Point (ICP) algorithm [18,19] alternates between correspondence assignment and transform estimation for point cloud alignment. Procrustes analysis [20] provides closed-form solutions for rigid alignment given correspondences. Extensions include weighted variants [21,22] and robust formulations. Our Stage 1 adapts ICP-style alternating optimization to the VPR similarity objective with provable monotone improvement over evaluated candidates.

**Robust Estimation.** Huber's M-estimators [23] downweight outliers in regression. Pose graph optimization [24,25] fuses odometry and visual constraints. Switchable constraints [26] and graduated non-convexity [27] handle outliers in SLAM. Our Stage 3 performs z-score outlier detection and *clamps* detected outliers to the VIO prior (by setting $\alpha_i \to \infty$), yielding a closed-form strictly convex solve without iterative robust optimization.

**Visual-Inertial Odometry.** VIO methods—including EKF-based VIO [28], OKVIS [29], VINS-Mono [30], and ORB-SLAM3 [31] —provide accurate *relative* motion estimation but

accumulate unbounded drift over long trajectories. These methods solve a fundamentally different problem than NaviLoc: VIO estimates how the platform moved (relative poses), while NaviLoc estimates where the platform is in the world (global positions). NaviLoc is designed to operate *on top of* any VIO system, using its relative motion output as a prior while correcting global drift via satellite imagery matching.

## 3. Method

This section presents the NaviLoc algorithm in three stages, illustrated in Figure 2. We first formalize the problem (Section 3.1) and then describe each stage: Global Align (Section 3.2) finds an initial SE(2) transform via robust trajectory-level optimization; Refinement (Section 3.3) applies windowed geometric corrections; and Smoothing (Section 3.4) fuses VIO constraints with VPR anchors while rejecting outliers. Each stage has a well-defined objective function with a closed-form or provably convergent solution.

### *3.1. Problem Formulation*

Given $N$ aerial query images with descriptors $\mathbf{F}_q = \{\mathbf{f}_i^q\}_{i=1}^N$ and VIO-derived positions $\mathbf{V} = \{(v_i^x, v_i^y)\}_{i=1}^N$ in a local frame, along with VIO displacements $\Delta\mathbf{V} = \{\mathbf{v}_{i+1} - \mathbf{v}_i\}$, and a geo-referenced satellite map with $M$ reference tiles having descriptors $\mathbf{F}_r = \{\mathbf{f}_j^r\}_{j=1}^M$ and coordinates $\mathbf{G} = \{(g_j^x, g_j^y)\}_{j=1}^M$, we seek global positions $\mathbf{P} = \{(p_i^x, p_i^y)\}_{i=1}^N$.

### *3.2. Stage 1: Global Align*

We estimate a global SE(2) transform $(\theta, \mathbf{t})$ by maximizing the trajectory-level similarity objective:

$$J(\theta, \mathbf{t}) = \frac{1}{N} \sum_{i=1}^{N} \max_{j \in \mathcal{B}(R_\theta \mathbf{v}_i + \mathbf{t}, r)} \langle \mathbf{f}_i^q, \mathbf{f}_j^r \rangle \tag{1}$$

where $R_\theta$ is the 2D rotation matrix, $\mathcal{B}(\mathbf{x}, r) = \{j : \|\mathbf{g}_j - \mathbf{x}\| \leq r\}$ is the set of reference tiles within radius $r$, and $\langle \cdot, \cdot \rangle$ denotes cosine similarity.

**Algorithm and intuition.** Stage 1 treats VPR as a noisy correspondence generator: for a fixed rotation $\theta$, each frame proposes a translation "measurement" $\mathbf{d}_i(\theta) = \mathbf{g}_{j_i^*} - R_\theta \mathbf{v}_i$ based on the global top-1 match $j_i^*$. Under perceptual aliasing, many $j_i^*$ are outliers; we therefore aggregate $\{\mathbf{d}_i(\theta)\}$ with a robust location estimator (the component-wise median). The key contribution is casting these noisy retrieval outcomes into a trajectory-level objective. We use coordinate ascent: (1) scan a grid of $K$ rotation angles $\theta \in \{-\pi, -\pi + 2\pi/K, \ldots\}$; (2) for each $\theta$, compute the L1-optimal translation (component-wise median) [23]:

$$\mathbf{t}(\theta) = \text{median}\{\mathbf{g}_{j_i^*} - R_\theta \mathbf{v}_i\}_{i=1}^N \tag{2}$$

where $j_i^* = \arg\max_j \langle \mathbf{f}_i^q, \mathbf{f}_j^r \rangle$ is the global top-1 match; (3) evaluate $J(\theta, \mathbf{t}(\theta))$ and select the best; and (4) refine with alternating maximization using local targets.

We denote the resulting aligned trajectory by $\mathbf{P}^{(1)} = \{\mathbf{p}_i^{(1)}\}_{i=1}^N$. The following result formalizes why the median aggregator is optimal for robust translation estimation under outlier contamination:

**Theorem 1 (L1-Optimal Translation (Median)).** *Let* $\mathbf{d}_i = \mathbf{g}_{j_i} - R_\theta \mathbf{v}_i \in \mathbb{R}^2$ *be translation residuals for fixed* $\theta$*. The minimizer of*

$$\min_{\mathbf{t} \in \mathbb{R}^2} \sum_{i=1}^{N} \|\mathbf{d}_i - \mathbf{t}\|_1 \tag{3}$$

*is given by the component-wise median:* $t_x = \text{median}\{d_{i,x}\}$ *and* $t_y = \text{median}\{d_{i,y}\}$. *Consequently,* $\mathbf{t}(\theta)$ *is robust to outlier residuals: as long as more than half of the residuals are inliers per coordinate, the estimate is controlled by the inlier set rather than the outliers.*

**Proof.** The L1 norm $\|\mathbf{d}_i - \mathbf{t}\|_1 = |d_{i,x} - t_x| + |d_{i,y} - t_y|$ decomposes across coordinates, so the objective separates into two independent 1D problems: $\min_{t_x} \sum_i |d_{i,x} - t_x|$ and $\min_{t_y} \sum_i |d_{i,y} - t_y|$. The minimizer of $\sum_i |x_i - t|$ over $t \in \mathbb{R}$ is the median of $\{x_i\}$ [23]. Robustness follows because the median is unaffected by arbitrarily large perturbations to fewer than half of the samples. □

**Remark 1 (monotone acceptance).** *In our implementation, we only accept candidate updates that improve (or tie)* $J$*, so the sequence of evaluated objective values is monotone non-decreasing, bounded above by* 1*, and hence convergent. This guarantees convergence of the objective sequence, not global optimality (see Algorithm A1).*

### *3.3. Stage 2: Refinement*

After global alignment, we refine positions using sliding-window bounded Procrustes. This extends standard weighted Procrustes analysis [20,21] with a novel rotation bound constraint that prevents overcorrection from noisy VPR targets. For each frame index $j$, we compute a *local VPR target* $\mathbf{t}_j \in \mathbb{R}^2$ as the coordinate of the best-matching reference tile within radius $r$ of the current predicted position $\mathbf{p}_j^{(1)}$ and record its cosine similarity score $s_j$. For each window $\mathcal{W} = \{i, i+1, \ldots, i+W-1\}$ with targets $\{\mathbf{t}_j\}_{j\in\mathcal{W}}$ and weights $w_j = \max(0, s_j)^2$, we solve:

$$\min_{R\in SO(2),\mathbf{t}} \sum_{j\in\mathcal{W}} w_j \|R\mathbf{p}_j^{(1)} + \mathbf{t} - \mathbf{t}_j\|^2, \quad \text{s.t. } |\theta| \le \theta_{\max} \tag{4}$$

This weighted Procrustes problem admits a closed-form solution for both rotation and translation. The rotation constraint prevents overcorrection when local VPR targets are noisy:

**Theorem 2 (Optimal Bounded Procrustes).** *For the constrained problem (4) in 2D, let* $\bar{\mathbf{p}} = \sum_j w_j \mathbf{p}_j^{(1)} / \sum_j w_j$ *and* $\bar{\mathbf{t}} = \sum_j w_j \mathbf{t}_j / \sum_j w_j$ *be the weighted centroids, and let* $H = \sum_{j\in\mathcal{W}} w_j (\mathbf{p}_j^{(1)} - \bar{\mathbf{p}})(\mathbf{t}_j - \bar{\mathbf{t}})^\top$ *be the* $2 \times 2$ *weighted cross-covariance matrix. The optimal rotation angle is* $\theta^* = \text{clip}(\hat{\theta}, -\theta_{\max}, \theta_{\max})$*, where* $\hat{\theta} = \text{atan2}(H_{10} - H_{01}, H_{00} + H_{11})$*, and the optimal translation is* $\mathbf{t}^* = \bar{\mathbf{t}} - R_{\theta^*}\bar{\mathbf{p}}$.

**Proof.** After centering by the weighted centroids, the objective separates into rotation and translation terms. For the rotation, we seek to maximize $\text{tr}(R_\theta H)$. Writing $R_\theta = \left(\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right)$, the trace expands to $(H_{00} + H_{11})\cos\theta + (H_{10} - H_{01})\sin\theta$ [20,21]. This sinusoidal function of $\theta$ is maximized at $\hat{\theta} = \text{atan2}(H_{10} - H_{01}, H_{00} + H_{11})$. The constraint $|\theta| \le \theta_{\max}$ clips this to the boundary if $\hat{\theta}$ lies outside. The optimal translation follows as $\mathbf{t}^* = \bar{\mathbf{t}} - R_{\theta^*}\bar{\mathbf{p}}$. □

Windows overlap (stride $S < W$), so each frame receives corrections from multiple windows; we average these contributions. Multiple passes are performed because each pass updates the trajectory, which changes the local VPR targets $\{\mathbf{t}_j\}$ (since they depend on the current predicted positions). Empirically, 2–3 passes suffice for convergence (see Algorithm A2).

### *3.4. Stage 3: Smoothing*

Let $\mathbf{P}^{(2)} = \{\mathbf{p}_i^{(2)}\}_{i=1}^N$ denote the Stage 2 output. We treat these positions as *anchors* $\mathbf{A} = \{\mathbf{a}_i\}_{i=1}^N$ with $\mathbf{a}_i := \mathbf{p}_i^{(2)}$ and fuse them with VIO constraints via a standard MAP formulation [32]. Our novel element is the z-score-based outlier detection that automatically clamps unreliable anchors to the VIO prior:

$$\min_{\mathbf{P}} \|\mathbf{DP} - \Delta\mathbf{V}\|^2 + \sum_{i=1}^{N} \alpha_i \|\mathbf{p}_i - \mathbf{a}_i\|^2 \tag{5}$$

where $\mathbf{D}$ is the $(N-1) \times N$ first-difference matrix.

**Outlier detection.** For each anchor $\mathbf{a}_i$, we recompute its local VPR similarity $s_i$ as the best cosine similarity among reference tiles within radius $r$ of $\mathbf{a}_i$. We then compute z-scores $z_i = (s_i - \bar{s})/\sigma_s$. Frames with $z_i < \tau$ (default $\tau = -1.5$) are marked as outliers and assigned $\alpha_i \to \infty$ (implemented as $10^6$), effectively clamping them to the VIO prior. Inliers use $\alpha_i = \alpha_{\text{in}} = 0.05$. With the anchor weights thus defined, the optimization problem has a unique closed-form solution:

**Theorem 3 (Unique Solution).** *If $\alpha_i > 0$ for at least one $i$, then the objective (5) is strictly convex with unique minimizer given by the linear system:*

$$(\mathbf{D}^\top\mathbf{D} + \text{diag}(\boldsymbol{\alpha}))\mathbf{P} = \mathbf{D}^\top\Delta\mathbf{V} + \boldsymbol{\alpha} \odot \mathbf{A} \tag{6}$$

**Proof.** The objective (5) is quadratic in $\mathbf{P}$. Setting the gradient to zero yields the normal equations $(\mathbf{D}^\top\mathbf{D} + \text{diag}(\boldsymbol{\alpha}))\mathbf{P} = \mathbf{D}^\top\Delta\mathbf{V} + \boldsymbol{\alpha} \odot \mathbf{A}$. The matrix $\mathbf{D}^\top\mathbf{D}$ is the graph Laplacian of a path, which is positive semidefinite with nullspace spanned by the constant vector $\mathbf{1}$. Adding $\text{diag}(\boldsymbol{\alpha})$ with at least one $\alpha_i > 0$ eliminates this nullspace: any $\mathbf{x} = c\mathbf{1}$ has $\mathbf{x}^\top\text{diag}(\boldsymbol{\alpha})\mathbf{x} = c^2\sum_i \alpha_i > 0$. Thus the combined matrix is positive definite, guaranteeing strict convexity and a unique solution [32]. □

See Algorithm A3 for pseudocode.

### *3.5. Implementation Details*

Algorithm 1 summarizes the complete pipeline. We use DeiT-Tiny-Distilled [33], a distilled Vision Transformer [34], for feature extraction (192-dim descriptors). Fixed parameters: search radius $r = 150$ m, $K = 72$ angles, window size $W = 10$, stride $S = 7$, $\theta_{\max} = 0.09$ rad (≈5°), and $z$-threshold $\tau = -1.5$.

**Algorithm 1** NaviLoc

**Require:** Query features $\mathbf{F}_q \in \mathbb{R}^{N\times D}$, VIO trajectory $\mathbf{V} \in \mathbb{R}^{N\times 2}$
**Require:** Reference database $(\mathbf{F}_r, \mathbf{G})$ with $M$ tiles
**Ensure:** Localized positions $\mathbf{P} \in \mathbb{R}^{N\times 2}$
$(\mathbf{P}, \theta^*) \leftarrow \text{GLOBALALIGN}(\mathbf{V}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G})$ ▷ Stage 1
$\mathbf{P} \leftarrow \text{REFINEMENT}(\mathbf{P}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G})$ ▷ Stage 2
$\mathbf{P}^* \leftarrow \text{SMOOTHING}(\mathbf{P}, \mathbf{V}, \theta^*, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G})$ ▷ Stage 3
**return** $\mathbf{P}^*$

Detailed pseudocode for each stage is provided in Appendix A.

## 4. Experiments

### *4.1. Dataset*

To the best of our knowledge, there is no publicly available *challenging* dataset for our target setting—low-altitude (50–150 m) UAV imagery with synchronized VIO and a

geo-referenced satellite tile database for trajectory-level evaluation—in particular with long real-world trajectories. We therefore evaluate on a prepared real-world UAV-to-satellite benchmark collected over rural terrain in Ukraine. Table 1 summarizes the dataset statistics. The dataset comprises 58 aerial query frames captured at 50–150 m AGL along a 2.3 km trajectory with 40 m inter-frame spacing. Reference imagery consists of 462 geo-referenced satellite tiles at 0.3 m/px resolution covering 1.6 km$^2$ with 40 m tile spacing. The benchmark is challenging due to strong perceptual aliasing in rural/village scenes (repetitive texture, limited distinctive landmarks) combined with low-altitude viewpoint and appearance changes relative to the satellite map.

**Data collection and preparation.** The query stream was recorded from a real multirotor UAV flight over rural/village terrain in Ukraine under summer conditions. The platform was manually piloted in an FPV-style flight mode at 50–150 m AGL, with average speed below 60 m/s, producing a 2.3 km trajectory.

All onboard processing was performed on a Raspberry Pi 5 (8 GB RAM). Images were captured by a rigidly mounted nadir-view Sony IMX219 RGB camera at 30 FPS with approximately 77–80° diagonal field of view, with auto-exposure and auto-white-balance enabled; a separate forward/oblique camera was used only for piloting and is not part of the evaluation. Inertial measurements were provided by the onboard SpeedyBee V3 IMU. Relative motion was estimated by a standard EKF-based monocular Visual-Inertial Odometry (VIO) pipeline fusing the camera and IMU. These components were chosen for availability, low cost, and suitability for lightweight embedded UAV platforms; NaviLoc is designed to work with any standard VIO pipeline and is agnostic to the specific sensor choice. As expected, VIO is subject to drift over extended flights and performs no global map alignment; NaviLoc uses only the resulting relative pose increments and is agnostic to the specific VIO implementation.

Ground truth positions are obtained from onboard GNSS logs and are used only for evaluation (MLE/ATE) and visualization; GNSS data are not available to NaviLoc during inference. The reference database was built by sampling satellite map imagery via the Google Maps API at zoom level 19 on a 40 m grid over the flight area; each tile is associated with its geographic coordinate. We then convert both query frames and reference tiles into embeddings using the backbones in Table 1.

**Table 1.** Dataset statistics.

| Property | Value |
|---|---|
| Query frames | 58 |
| Query spacing | 40 m |
| Trajectory length | 2323 m |
| Flight altitude (AGL) | 50–150 m |
| Reference tiles | 462 |
| Reference tile spacing | 40 m |
| Reference coverage | 1.6 km$^2$ |

### *4.2. Baselines*

We compare against:

- **Raw VIO (IP only)**: VIO trajectory with initial position aligned to ground truth (no rotation).
- **VIO + IP + IR (oracle)**: VIO with initial position and initial rotation estimated from the first displacement (one-segment oracle) and from the first $k = 10$ displacements (multi-segment oracle).
- **VIO SE(2) oracle**: VIO with oracle global SE(2) alignment to ground truth (best possible rigid alignment).

- **Per-frame VPR (top-*k*)**: For each query frame, retrieve the *k* reference tiles with highest cosine similarity and predict position as their coordinate mean (top-1 uses the single best match; top-3 averages the three best).
- **AnyLoc-VLAD** [1]: State-of-the-art VPR using DINOv2 [35] with VLAD aggregation, evaluated with top-3 retrieval.

### *4.3. Results*

On this challenging real-world benchmark, NaviLoc achieves **19.5 m** MLE—a **16.0**× improvement over the state-of-the-art AnyLoc-VLAD, and a **32.1**× improvement over raw VIO drift. Figures 3 and 4 summarize these results.

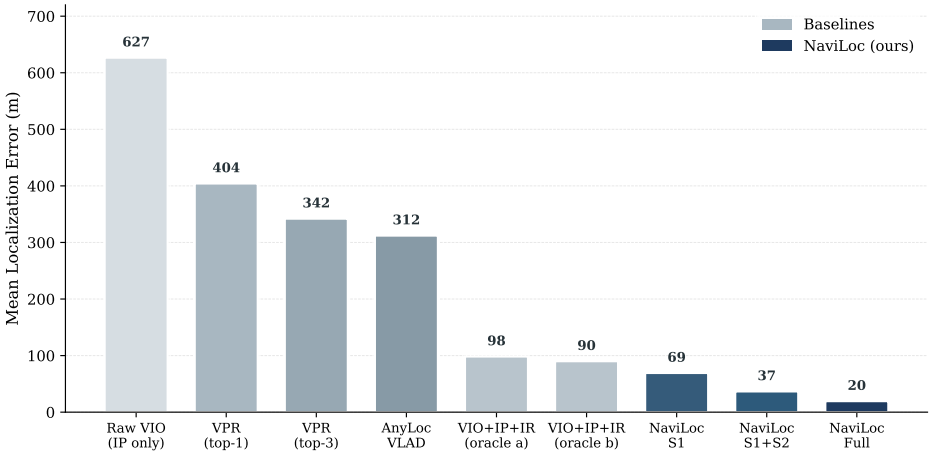



**Figure 3.** Mean Localization Error comparison. NaviLoc (19.5 m) outperforms all baselines by a wide margin.

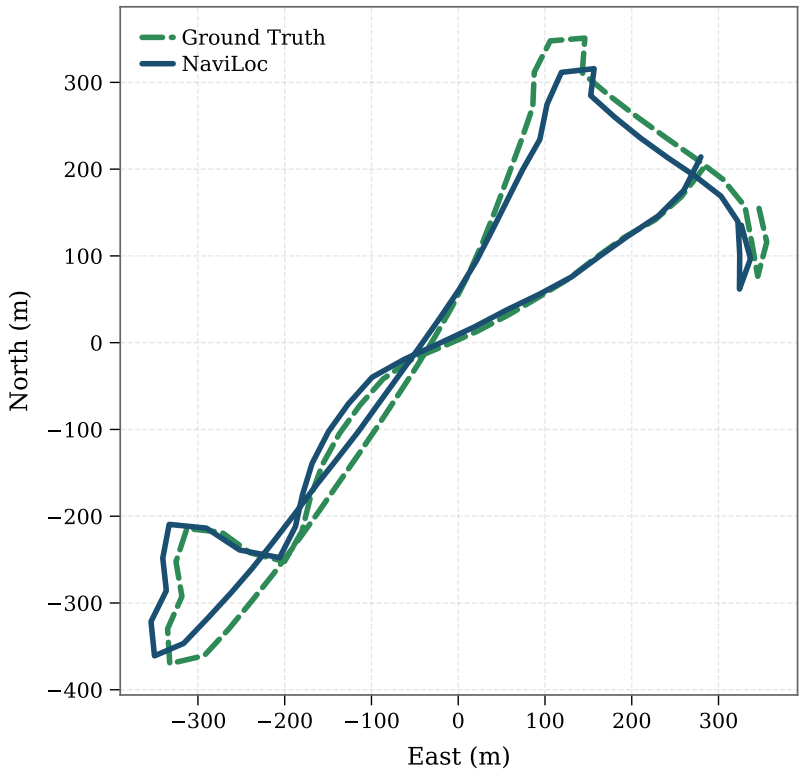



**Figure 4.** Trajectory comparison: ground truth (green, dashed) versus NaviLoc estimate (navy, solid). NaviLoc correctly recovers the global trajectory despite per-frame VPR noise.

### *4.4. Ablation Studies*

**Stage contributions.** Each stage provides substantial improvement (Table 2). Stage 1 alone achieves 9× improvement by finding the correct global transform. Stage 2 refines local errors, and Stage 3 leverages VIO constraints while excluding outliers.

**Outlier threshold.** Table 3 shows sensitivity to the z-score threshold.

**Table 2.** Quantitative comparison. MLE = Mean Localization Error; ATE = Absolute Trajectory Error.

| Method | MLE (m) | ATE (m) | Improvement |
|---|---|---|---|
| Raw VIO (IP only) | 626.7 | 728.9 | 1.0× |
| VIO + IP + IR (first delta, oracle) * | 98.4 | 132.5 | 6.4× |
| VIO + IP + IR (first $k = 10$ deltas, oracle) * | 90.1 | 132.2 | 7.0× |
| Per-frame VPR (top-1) | 404.3 | 471.8 | 1.5× |
| Per-frame VPR (top-3) | 342.1 | 413.8 | 1.8× |
| AnyLoc-VLAD (top-3) [1] | 312.2 | 368.5 | 2.0× |
| VIO SE(2) oracle * | 54.2 | 67.9 | 11.6× |
| NaviLoc: Stage 1 | 69.3 | 76.9 | 9.0× |
| NaviLoc: Stages 1 + 2 | 36.7 | 42.6 | 17.1× |
| **NaviLoc: Full** | **19.5** | **21.6** | **32.1**× |

* Oracle baselines use ground truth information.

**Table 3.** Outlier detection threshold ablation.

| Threshold | Outliers | MLE (m) | Improvement |
|---|---|---|---|
| None | 0 | 26.6 | 23.5× |
| $z < -2.0$ | 3 | 21.4 | 29.3× |
| $z < -1.5$ | 7 | 19.5 | 32.1× |
| $z < -1.0$ | 9 | 27.2 | 23.1× |

**Hyperparameter sensitivity.** Table 4 varies one hyperparameter at a time (others fixed to defaults). Performance is stable across moderate changes, while overly permissive local search radii or underconstrained refinement settings degrade accuracy.

**Table 4.** One-at-a-time hyperparameter sensitivity (full NaviLoc pipeline). Default settings are bold.

| Hyperparameter | Setting | MLE (m) |
|---|---|---|
| Rotation grid ($K$ angles) | 36/**72**/144 | 32.7/**19.5**/23.8 |
| Local search radius $r$ (m) | 100/**150**/200 | 36.6/**19.5**/83.0 |
| Stage 1 iterations | 1/2/**3**/4 | 19.8/22.2/**19.5**/19.5 |
| Window size $W$ (frames) | 9/**10**/11/12 | 27.6/**19.5**/24.6/27.2 |
| Stride $S$ (frames) | 5/**7**/9 | 33.2/**19.5**/35.2 |
| Rotation bound $\theta_{\max}$ (rad) | 0.05/**0.09**/0.12 | 24.3/**19.5**/19.2 |
| Anchor weight $\alpha_{\text{in}}$ | 0.02/**0.05**/0.10 | 20.6/**19.5**/22.6 |

**Backbone comparison.** Table 5 reveals that DeiT-Tiny-Distilled achieves the best NaviLoc performance despite not having the best per-frame VPR accuracy. Empirically, distillation appears to yield descriptors whose scores are more consistent *across time*, making trajectory-level alignment easier.

**Table 5.** Backbone comparison.

| Backbone | Dimensions | VPR MLE (m) | NaviLoc MLE (m) |
|---|---|---|---|
| DeiT-Tiny-Distilled | 192 | 342.1 | **19.5** |
| ConvNeXt-Tiny | 768 | 364.1 | 42.0 |
| LeViT-256 (distilled) | 512 | 346.7 | 70.6 |
| DeiT-Tiny | 192 | 430.4 | 105.1 |
| DINOv2-ViT-G/14 + VLAD | 98,304 | 356.7 | 162.0 |

### *4.5. Computational Efficiency*

NaviLoc operates on precomputed descriptors. We benchmark on Raspberry Pi 5 (ARM Cortex-A76, 8 GB RAM). Table 6 summarizes the computational performance.

**Table 6.** Computational performance on Raspberry Pi 5.

| Component | Time (ms) | FPS |
|---|---|---|
| Feature Extraction (DeiT-Tiny) | 79 | 12.7 |
| NaviLoc Algorithm (C++) | 32 | 31 |
| **End-to-End Inference** | **111** | **9.0** |

**Inference strategy.** At each timestep, the system samples an aerial image, extracts a 192-dimensional DeiT-Tiny-Distilled embedding (79 ms), and updates the global trajectory estimate using the C++ NaviLoc algorithm (32 ms). The combined end-to-end latency of **111 ms** yields **9.0 FPS** on Raspberry Pi 5, enabling real-time embedded deployment. Feature extraction becomes the bottleneck; DeiT-Tiny-Distilled requires 1.3 GFLOPs per image versus 21.1 GFLOPs for DINOv2-S, providing 16× faster extraction while achieving superior localization accuracy.

## 5. Discussion

We now analyze NaviLoc's performance across multiple dimensions, compare it with alternative approaches, and discuss limitations and future directions.

### *5.1. Comparison with Existing Methods*

**Versus per-frame VPR.** Per-frame VPR methods treat each query independently, producing 342.1 m MLE (top-3) on our benchmark. NaviLoc achieves 19.5 m—a **17.5× improvement**. The key insight is that perceptual aliasing causes individual VPR errors that are spatially inconsistent: incorrect matches scatter across the map, while correct matches cluster near the true trajectory. Trajectory-level optimization exploits this statistical property.

**Versus state-of-the-art.** AnyLoc [1] represents the state-of-the-art in universal VPR, aggregating DINOv2 [35] features via VLAD pooling. On our benchmark, AnyLoc-VLAD achieves 312.2 m MLE (top-3)—only marginally better than simple per-frame VPR. NaviLoc reduces this to 19.5 m, a **16.0× improvement**. This demonstrates that even powerful foundation model features remain susceptible to cross-view aliasing without trajectory-level reasoning.

**Versus VIO baselines.** Raw VIO accumulates 626.7 m drift over the 2.3 km trajectory. Even with oracle initial rotation alignment, VIO achieves only 90.1 m MLE due to scale and heading drift. NaviLoc's 19.5 m MLE represents a **32.1× improvement** over raw VIO, highlighting the benefit of trajectory-level use of noisy VPR measurements to correct long-horizon drift.

**Versus graph-based SLAM.** Traditional pose graph optimization [24,25] formulates localization as nonlinear least squares over loop closure constraints. These methods typically require multiple sensors (LiDAR, stereo cameras, IMU), 3D point cloud processing, and GPU acceleration for real-time operation. NaviLoc is fundamentally lighter: it operates on a single monocular camera plus VIO, uses 2D tile descriptors rather than 3D geometry, and requires no GPU. Each stage employs closed-form solutions (median, SVD, linear solve) rather than iterative solvers (Gauss-Newton, Levenberg-Marquardt) and handles outliers via z-score clamping rather than iterative robust optimization [26,27]. This yields deterministic, bounded runtime on CPU-only embedded platforms.

### *5.2. Computational Efficiency*

**Real-time embedded deployment.** NaviLoc achieves **9 FPS end-to-end on Raspberry Pi 5**, combining 79 ms feature extraction (DeiT-Tiny-Distilled) with 32 ms trajectory optimization (C++). This demonstrates that trajectory-level cross-view localization can run in real time on a CPU-only embedded platform.

**Comparison with foundation model approaches.** While larger backbones can improve standalone VPR on some benchmarks, they are substantially more expensive for embedded CPU deployment. Our ablation (Table 5) shows that DINOv2-ViT-G/14+VLAD achieves 162.0 m MLE versus 19.5 m for DeiT-Tiny-Distilled—larger models did not improve NaviLoc on this cross-view dataset.

**Runtime scaling.** Let $N$ denote trajectory length and $M$ reference database size. Stage 1 performs $O(N)$ global nearest-neighbor queries, each requiring $O(M)$ comparisons, yielding $O(NM)$. The angle grid size and iteration counts are small constants. Stage 2 runs in $O(N)$ per pass with constant window size. Stage 3 solves a tridiagonal system in $O(N)$ via the Thomas algorithm [36]. The overall time complexity is $O(NM)$; in practice, feature extraction dominates runtime. For larger reference databases, approximate nearest-neighbor structures could reduce this to sublinear in $M$.

### *5.3. Portability and Transferability*

**Training-free operation.** Unlike learned cross-view methods [7,8] that require domain-specific training data, NaviLoc operates on pretrained features without fine-tuning. This is critical for GNSS-denied scenarios where ground truth correspondences are unavailable for training.

**Backbone flexibility.** NaviLoc accepts any image descriptor; our ablation tested five backbones (Table 5). This modularity allows leveraging future feature extractors without algorithmic changes.

**Simplicity.** NaviLoc has no learned components beyond the feature extractor, no hyperparameter schedules, and no complex data association. This facilitates debugging and deployment in safety-critical applications.

### *5.4. Robustness Analysis*

**Hyperparameter stability.** Table 4 demonstrates that NaviLoc maintains sub-25 m accuracy across $\pm 50\%$ perturbations of most hyperparameters. The outlier threshold (Table 3) shows graceful degradation outside the optimal range. This robustness simplifies deployment: practitioners can use default settings without extensive tuning.

**Adaptive outlier detection.** Z-score normalization adapts to per-trajectory statistics: a similarity of 0.25 may indicate an outlier in one flight but a reliable match in another. This eliminates per-dataset threshold tuning.

### *5.5. Feature Descriptor Analysis*

Table 5 shows that DeiT-Tiny-Distilled (5M parameters) achieves 19.5 m MLE, while DINOv2-ViT-G/14 (1.1B parameters) achieves 162.0 m—**8**$\times$ **worse** despite being 200$\times$ larger. This empirical finding suggests that model selection for trajectory-level VPR should consider compatibility with geometric reasoning, not just per-frame retrieval accuracy.

### *5.6. Practical Extensions*

**Long-trajectory operation.** The current evaluation uses a 2.3 km trajectory processed in batch. For arbitrarily long flights, NaviLoc can be extended via a sliding-window approach: maintain a context window of the most recent 2–3 km of trajectory data and

apply NaviLoc to this window for each new position estimate. This bounds memory and computation while providing continuous global localization.

### *5.7. Limitations and Future Work*

**VIO dependency.** NaviLoc assumes VIO provides accurate relative motion over short horizons. Significant VIO failures (e.g., from aggressive maneuvers or texture-poor environments) would propagate to the final estimate. Future work could jointly estimate VIO scale and bias.

**Trajectory length.** Very short trajectories (<10 frames) provide insufficient statistics for robust similarity aggregation. A minimum flight distance of approximately 400 m is recommended. Conversely, NaviLoc handles arbitrarily long trajectories via the sliding-window extension described above.

**Seasonal and environmental variation.** Our evaluation uses satellite imagery captured under similar conditions (summer, clear weather) to the query flight. Performance under adverse weather conditions (rain, fog, and low visibility), seasonal changes (snow cover, vegetation differences), or significant lighting variations between the reference map and query images remains unevaluated. The training-free design may offer some robustness since the DeiT feature extractor was trained on diverse ImageNet conditions, but systematic evaluation across weather and seasonal variations is an important direction for future work.

**Additional trajectories.** Due to the complexity of real-world UAV data collection with synchronized VIO and ground truth, our evaluation is limited to a single 2.3 km trajectory. While the ablation studies (Tables 3–5) provide statistical validation across hyperparameter variations, the evaluation of additional trajectories in diverse environments would further strengthen the generalization claims. Collecting and evaluating datasets spanning multiple environments is planned as future work.

**3D extension.** The current SE(2) formulation assumes approximately planar flight. Extending to SE(3) would support altitude variations and complex terrain, potentially leveraging 3D point cloud or mesh references.

**Online operation.** NaviLoc currently processes trajectories in batch. An incremental variant that updates estimates as new frames arrive would enable tighter integration with flight controllers for real-time autonomous navigation.

**Energy efficiency.** Reducing onboard computation energy extends flight endurance and lowers environmental impact. Systematic energy profiling [37] could guide model selection and duty-cycling to further improve efficiency.

## 6. Conclusions

We presented NaviLoc, a three-stage trajectory-level localization pipeline with rigorous mathematical foundations. Each stage has a well-defined objective and provable convergence properties. NaviLoc achieves 19.5 m Mean Localization Error on a UAV-to-satellite benchmark, representing a **16× improvement** over AnyLoc-VLAD, the state-of-the-art VPR method. End-to-end inference runs at **9 FPS** on Raspberry Pi 5, demonstrating practical viability for real-time embedded UAV navigation.

**Author Contributions:** Conceptualization, methodology, software, experiments, visualization, and writing—original draft: P.S.; supervision, formal analysis, writing—review and editing, and funding acquisition: T.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by Academia Tech, Taras Shevchenko National University of Kyiv, and Hackathon Expert NGO.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The experimental data and code used in this study are available from the corresponding author upon reasonable request for non-commercial academic research purposes. Requests must be submitted from an academic email address; access is granted after signing an NDA. Eligible requesters will receive a link to a private GitHub (https://github.com/) repository and will be added with read-only access.

**Acknowledgments:** The authors thank Academia Tech for providing datasets and computational infrastructure to support this research.

**Conflicts of Interest:** Author Pavel Shpagin is employed by Academia Tech (Research and Development Division, Kyiv, Ukraine). The remaining author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| AGL | Above Ground Level |
| ATE | Absolute Trajectory Error |
| CPU | Central Processing Unit |
| EKF | Extended Kalman Filter |
| FPS | Frames Per Second |
| GNSS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| GPU | Graphics Processing Unit |
| ICP | Iterative Closest Point |
| IMU | Inertial Measurement Unit |
| LiDAR | Light Detection and Ranging |
| MAP | Maximum A Posteriori |
| MLE | Mean Localization Error |
| MSCKF | Multi-State Constraint Kalman Filter |
| OKVIS | Open Keyframe-based Visual-Inertial SLAM |
| RAM | Random Access Memory |
| SE(2) | Special Euclidean Group in 2D |
| SLAM | Simultaneous Localization and Mapping |
| SVD | Singular Value Decomposition |
| UAV | Unmanned Aerial Vehicle |
| VIO | Visual-Inertial Odometry |
| ViT | Vision Transformer |
| VLAD | Vector of Locally Aggregated Descriptors |
| VPR | Visual Place Recognition |

## Appendix A. Detailed Algorithm Pseudocode

This appendix provides detailed pseudocode for each stage of NaviLoc with complete mathematical specifications.

### *Appendix A.1. Stage 1: Global Align*

**Algorithm A1** Global Align (Stage 1)

**Require:** VIO trajectory $\mathbf{V} \in \mathbb{R}^{N\times 2}$, Query features $\mathbf{F}_q \in \mathbb{R}^{N\times D}$
**Require:** Reference features $\mathbf{F}_r \in \mathbb{R}^{M\times D}$, Reference coords $\mathbf{G} \in \mathbb{R}^{M\times 2}$
**Require:** Number of angles $K = 72$, Search radius $r$, AM iterations $I_{\max} = 3$
**Ensure:** Aligned trajectory $\mathbf{P}$, Rotation angle $\theta^*$

▷ Phase 1: Coarse grid search over rotation

1: $\Theta \leftarrow \{-\pi + 2\pi k/K : k = 0, \ldots, K-1\}$

**Algorithm A1** *Cont.*

$\mathbf{T}_{\text{global}} \leftarrow \text{GlobalTop1}(\mathbf{F}_q, \mathbf{F}_r, \mathbf{G})$ ▷ VPR targets
$J^* \leftarrow -\infty$, $\theta^* \leftarrow 0$
**for** $\theta \in \Theta$ **do**
  $\mathbf{V}_\theta \leftarrow R_\theta \mathbf{V}$ ▷ Rotate VIO
  $\mathbf{t} \leftarrow \text{median}(\mathbf{T}_{\text{global}} - \mathbf{V}_\theta)$ ▷ L1-optimal translation
  $\mathbf{P} \leftarrow \mathbf{V}_\theta + \mathbf{t}$
  $J \leftarrow \text{Score}(\mathbf{P}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G}, r)$ ▷ Equation (1)
  **if** $J > J^*$ **then**
    $J^* \leftarrow J$, $\theta^* \leftarrow \theta$, $\mathbf{t}^* \leftarrow \mathbf{t}$
  **end if**
**end for**
$\mathbf{P} \leftarrow R_{\theta^*} \mathbf{V} + \mathbf{t}^*$
▷ Phase 2: Alternating maximization
**for** $i = 1, \ldots, I_{\max}$ **do**
  $\mathbf{T}_{\text{local}} \leftarrow \text{LocalTop1}(\mathbf{P}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G}, r)$
  $\delta \leftarrow 2\pi/K \cdot 2^{1-\min(i,2)}$ ▷ Shrinking grid
  $\Theta_{\text{local}} \leftarrow \{\theta^* + \delta \cdot (2k/35 - 1) : k = 0, \ldots, 35\}$
  improved $\leftarrow$ **false**
  **for** $\theta \in \Theta_{\text{local}}$ **do**
    $\mathbf{V}_\theta \leftarrow R_\theta \mathbf{V}$
    $\mathbf{t} \leftarrow \text{median}(\mathbf{T}_{\text{local}} - \mathbf{V}_\theta)$
    $\mathbf{P}_{\text{test}} \leftarrow \mathbf{V}_\theta + \mathbf{t}$
    $J \leftarrow \text{Score}(\mathbf{P}_{\text{test}}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G}, r)$
    **if** $J > J^* + \epsilon$ **then**
      $J^* \leftarrow J$, $\theta^* \leftarrow \theta$, $\mathbf{t}^* \leftarrow \mathbf{t}$, $\mathbf{P} \leftarrow \mathbf{P}_{\text{test}}$
      improved $\leftarrow$ **true**
    **end if**
  **end for**
  **if** ¬improved **then**
    **break** ▷ Converged
  **end if**
**end for**
**return** $\mathbf{P}$, $\theta^*$

*Appendix A.2. Stage 2: Refinement*

**Algorithm A2** Refinement (Stage 2)

**Require:** Trajectory $\mathbf{P} \in \mathbb{R}^{N\times 2}$, Features $\mathbf{F}_q$, Reference $(\mathbf{F}_r, \mathbf{G})$
**Require:** Window size $W = 10$, Stride $S = 7$, Max rotation $\theta_{\max} = 0.09$, Passes $P_{\max} = 3$
**Ensure:** Refined trajectory $\mathbf{P}$
**for** $p = 1, \ldots, P_{\max}$ **do**
  $\mathbf{A} \leftarrow \mathbf{0}_{N\times 2}$, $\mathbf{C} \leftarrow \mathbf{0}_N$ ▷ Accumulators
  **for** start $= 0, S, 2S, \ldots$ **while** start $< N$ **do**
    end $\leftarrow \min(\text{start} + W, N)$
    $\mathcal{W} \leftarrow \{\text{start}, \ldots, \text{end} - 1\}$ ▷ Window indices
    **if** $|\mathcal{W}| < 3$ **then continue**
    **end if**
    $(\mathbf{T}, \mathbf{s}) \leftarrow \text{LocalTop1WithSim}(\mathbf{P}_{\mathcal{W}}, \mathbf{F}_q[\mathcal{W}], \mathbf{F}_r, \mathbf{G}, r)$
    $\mathbf{w} \leftarrow \max(0, \mathbf{s})^2$ ▷ Squared similarity weights
    ▷ Bounded weighted Procrustes
    $\bar{\mathbf{p}} \leftarrow \sum_j w_j \mathbf{P}_j / \sum_j w_j$
    $\bar{\mathbf{t}} \leftarrow \sum_j w_j \mathbf{T}_j / \sum_j w_j$
    $H \leftarrow \sum_j w_j (\mathbf{P}_j - \bar{\mathbf{p}})(\mathbf{T}_j - \bar{\mathbf{t}})^\top$ ▷ $2 \times 2$ cross-covariance

**Algorithm A2** *Cont.*

$(U, \Sigma, V^\top) \leftarrow \mathrm{SVD}(H)$
$R \leftarrow VU^\top$
**if** $\det(R) < 0$ **then** $V[:,1] \leftarrow -V[:,1]$; $R \leftarrow VU^\top$
**end if**
▷ Clip rotation to bounded interval
$\theta \leftarrow \mathrm{atan2}(R_{10}, R_{00})$
$\theta \leftarrow \mathrm{clip}(\theta, -\theta_{\max}, \theta_{\max})$
$R_{\text{bounded}} \leftarrow \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$
$\mathbf{t}_{\text{bounded}} \leftarrow \bar{\mathbf{t}} - R_{\text{bounded}}\bar{\mathbf{p}}$
▷ Apply transform
$\mathbf{P}'_{\mathcal{W}} \leftarrow R_{\text{bounded}}\mathbf{P}_{\mathcal{W}} + \mathbf{t}_{\text{bounded}}$
$\mathbf{A}[\mathcal{W}] \leftarrow \mathbf{A}[\mathcal{W}] + \mathbf{P}'_{\mathcal{W}}$
$\mathbf{C}[\mathcal{W}] \leftarrow \mathbf{C}[\mathcal{W}] + 1$
**end for**
$\mathbf{C}[\mathbf{C} = 0] \leftarrow 1$ ▷ Avoid division by zero
$\mathbf{P} \leftarrow \mathbf{A}/\mathbf{C}$ ▷ Average overlapping estimates
**end for**
**return P**

*Appendix A.3. Stage 3: Smoothing*

**Algorithm A3** Smoothing (Stage 3)

**Require:** Anchor trajectory $\mathbf{A} \in \mathbb{R}^{N\times 2}$, VIO $\mathbf{V}$, Rotation $\theta$, Features $\mathbf{F}_q$
**Require:** Z-threshold $\tau = -1.5$, $\alpha_{\text{in}} = 0.05$
**Ensure:** Fused trajectory $\mathbf{P}^*$
$\mathbf{V}_\theta \leftarrow R_\theta \mathbf{V}$ ▷ Rotate VIO to global frame
$\Delta\mathbf{V} \leftarrow \mathrm{diff}(\mathbf{V}_\theta)$ ▷ $(N-1)\times 2$ displacements
▷ Compute local VPR similarities
$(_, \mathbf{s}) \leftarrow \mathrm{LocalTop1WithSim}(\mathbf{A}, \mathbf{F}_q, \mathbf{F}_r, \mathbf{G}, r)$
▷ Z-score outlier detection
$\bar{s} \leftarrow \mathrm{mean}(\mathbf{s})$, $\sigma_s \leftarrow \mathrm{std}(\mathbf{s})$
$\mathbf{z} \leftarrow (\mathbf{s} - \bar{s})/(\sigma_s + \epsilon)$
$\mathrm{outlier}_i \leftarrow (z_i < \tau)$ for $i = 1, \ldots, N$
▷ Set anchor weights (outliers clamped)
**for** $i = 1, \ldots, N$ **do**
$\alpha_i \leftarrow \begin{cases} 10^6 & \text{if outlier}_i \\ \alpha_{\text{in}} & \text{otherwise} \end{cases}$
**end for**
▷ Build and solve linear system
Construct $(N-1)\times N$ difference matrix $\mathbf{D}$: $D_{i,i} = -1$, $D_{i,i+1} = 1$
$\mathbf{M} \leftarrow \mathbf{D}^\top\mathbf{D} + \mathrm{diag}(\boldsymbol{\alpha})$ ▷ $N \times N$ positive definite
**for** $d \in \{x, y\}$ **do**
$\mathbf{b} \leftarrow \mathbf{D}^\top\Delta\mathbf{V}_d + \boldsymbol{\alpha} \odot \mathbf{A}_d$
$\mathbf{P}^*_d \leftarrow \mathrm{solve}(\mathbf{M}, \mathbf{b})$
**end for**
▷ Final uniform smoothing
$\boldsymbol{\alpha}_{\text{final}} \leftarrow \alpha_{\text{in}} \cdot \mathbf{1}_N$
$\mathbf{M}_{\text{final}} \leftarrow \mathbf{D}^\top\mathbf{D} + \mathrm{diag}(\boldsymbol{\alpha}_{\text{final}})$
**for** $d \in \{x, y\}$ **do**
$\mathbf{b} \leftarrow \mathbf{D}^\top\Delta\mathbf{V}_d + \alpha_{\text{in}} \cdot \mathbf{P}^*_d$
$\mathbf{P}^*_d \leftarrow \mathrm{solve}(\mathbf{M}_{\text{final}}, \mathbf{b})$
**end for**
**return** $\mathbf{P}^*$

## References


1. Keetha, N.; Mishra, A.; Karhade, J.; Jatavallabhula, K.M.; Scherer, S.; Krishna, M.; Garg, S. AnyLoc: Towards Universal Visual Place Recognition. *IEEE Robot. Autom. Lett.* **2024**, *9*, 1286–1293. [CrossRef]
2. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
3. Sivic, J.; Zisserman, A. Video Google: A Text Retrieval Approach to Object Matching in Videos. In Proceedings of the IEEE Conference on International Conference on Computer Vision (ICCV), Nice, France, 13–16 October 2003; pp. 1470–1477. [CrossRef]
4. Gálvez-López, D.; Tardós, J.D. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Trans. Robot.* **2012**, *28*, 1188–1197. [CrossRef]
5. Arandjelović, R.; Gronat, P.; Torii, A.; Pajdla, T.; Sivic, J. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 5297–5307. [CrossRef]
6. Ali-bey, A.; Chaib-draa, B.; Giguère, P. MixVPR: Feature Mixing for Visual Place Recognition. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2–7 January 2023; pp. 2998–3007. [CrossRef]
7. Hu, S.; Feng, M.; Nguyen, R.M.; Lee, G.H. CVM-Net: Cross-View Matching Network for Image-Based Ground-to-Aerial Geo-Localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7258–7267. [CrossRef]
8. Zhu, S.; Shah, M.; Chen, C. VIGOR: Cross-View Image Geo-localization beyond One-to-one Retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 3640–3649. [CrossRef]
9. Workman, S.; Souvenir, R.; Jacobs, N. Wide-Area Image Geolocalization with Aerial Reference Imagery. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3961–3969. [CrossRef]
10. Zheng, Z.; Wei, Y.; Yang, Y. University-1652: A Multi-view Multi-source Benchmark for Drone-based Geo-localization. In Proceedings of the ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 1395–1403. [CrossRef]
11. Ding, L.; Zhou, J.; Meng, L.; Long, Z. A Practical Cross-View Image Matching Method between UAV and Satellite for UAV-Based Geo-Localization. *Remote Sens.* **2021**, *13*, 47. [CrossRef]
12. Zhuo, X.; Koch, T.; Kurz, F.; Fraundorfer, F.; Reinartz, P. Automatic UAV Image Geo-Registration by Matching UAV Images to Georeferenced Image Data. *Remote Sens.* **2017**, *9*, 376. [CrossRef]
13. Zhuang, J.; Dai, M.; Chen, X.; Zheng, E. A Faster and More Effective Cross-View Matching Method of UAV and Satellite Images for UAV Geolocalization. *Remote Sens.* **2021**, *13*, 3979. [CrossRef]
14. Cui, Z.; Zhou, P.; Wang, X.; Zhang, Z.; Li, Y.; Li, H.; Zhang, Y. A Novel Geo-Localization Method for UAV and Satellite Images Using Cross-View Consistent Attention. *Remote Sens.* **2023**, *15*, 4667. [CrossRef]
15. Couturier, A.; Akhloufi, M.A. A Review on Deep Learning for UAV Absolute Visual Localization. *Drones* **2024**, *8*, 622. [CrossRef]
16. Zváriková, K.; Lázaro, A.; Durana, P. Neural Network-based Recognition and Virtual Simulation Algorithms, Interactive 3D Geo-Visualization and Neuromorphic Computing Systems, and Tactile Sensing and Cognitive Modeling Technologies in Web3-powered Metaverse Worlds. *Rev. Contemp. Philos.* **2023**, *22*, 170–186. [CrossRef]
17. Kurdel, P.; Lazar, T.; Mrekaj, B.; Hovanec, M.; Novák Sedláčková, A. Air Safety Information Management. In *Aviation Technologies, Systems and Research*; Springer: Cham, Switzerland, 2023; pp. 1–26. [CrossRef]
18. Besl, P.J.; McKay, N.D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239–256. [CrossRef]
19. Chen, Y.; Medioni, G. Object Modelling by Registration of Multiple Range Images. *Image Vis. Comput.* **1992**, *10*, 145–155. [CrossRef]
20. Schönemann, P.H. A Generalized Solution of the Orthogonal Procrustes Problem. *Psychometrika* **1966**, *31*, 1–10. [CrossRef]
21. Arun, K.S.; Huang, T.S.; Blostein, S.D. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *9*, 698–700. [CrossRef]
22. Umeyama, S. Least-Squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 376–380. [CrossRef]
23. Huber, P.J. *Robust Statistics*; John Wiley & Sons: New York, NY, USA, 1981; ISBN 978-0-471-41805-4.
24. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
25. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. g2o: A General Framework for Graph Optimization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011; pp. 3607–3613. [CrossRef]
26. Sünderhauf, N.; Protzel, P. Switchable Constraints for Robust Pose Graph SLAM. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, 7–12 October 2012; pp. 1879–1884. [CrossRef]

27. Yang, H.; Antonante, P.; Tzoumas, V.; Carlone, L. Graduated Non-Convexity for Robust Spatial Perception. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1127–1134. [CrossRef]
28. Mourikis, A.I.; Roumeliotis, S.I. A Multi-State Constraint Kalman Filter for Vision-Aided Inertial Navigation. In Proceedings of the IEEE International Conference on Robotics & Automation (ICRA), Rome, Italy, 10–14 April 2007; pp. 3565–3572. [CrossRef]
29. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]
30. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
31. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
32. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004; ISBN 978-0-521-83378-3.
33. Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; Jégou, H. Training Data-efficient Image Transformers & Distillation through Attention. In Proceedings of the International Conference on Machine Learning (ICML), Virtual, 18–24 July 2021; pp. 10347–10357.
34. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. In Proceedings of the International Conference on Learning Representations (ICLR), Vienna, Austria, 4 May 2021.
35. Oquab, M.; Darcet, T.; Moutakanni, T.; Vo, H.V.; Szafraniec, M.; Khalidov, V.; Fernandez, P.; Haziza, D.; Massa, F.; El-Nouby, A.; et al. DINOv2: Learning Robust Visual Features without Supervision. *arXiv* **2023**, arXiv:2304.07193.
36. Golub, G.H.; Van Loan, C.F. *Matrix Computations*, 4th ed.; Johns Hopkins University Press: Baltimore, MD, USA, 2013; ISBN 978-1-4214-0794-4. Available online: https://www.worldcat.org/isbn/9781421407944 (accessed on 20 November 2025).
37. Panchenko, T.V.; Piatygorskiy, N.D. Enrichment of the HEPscore Benchmark by Energy Consumption Assessment. *Technologies* **2025**, *13*, 362. [CrossRef]


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.