2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

BEV-Patch-PF: Particle Filtering with BEV-Aerial Feature Matching for Off-Road Geo-Localization

Dongmyeong Lee[1], Jesse Quattrociocchi[2], Christian Ellis[2], Rwik Rana[1], Amanda Adkins[1], Adam Uccello[2], Garrett Warnell[2], and Joydeep Biswas[1]

[1]University of Texas at Austin [2]DEVCOM Army Research Laboratory

https://amrl.cs.utexas.edu/bev-patch-pf

***Abstract*— Localizing ground robots against aerial imagery provides a critical capability for autonomous navigation, especially in environments where GPS is unreliable or unavailable. This task is challenging due to large viewpoint differences and substantial environmental variability. Most prior methods localize each frame independently, using either global-descriptor retrieval or spatial feature alignment, which leaves them vulnerable to ambiguity and multi-modal pose hypotheses. While sequential reasoning can mitigate this uncertainty, adapting existing per-frame pipelines for sequential use introduces unfavorable trade-offs among accuracy, memory, and computation that limit their practical deployment. We propose BEV-Patch-PF, a vision-only, GPS-free sequential geo-localization system that integrates particle filtering with learned bird's-eye-view (BEV) and aerial feature maps. For each 3-DoF particle pose hypothesis, we crop the corresponding patch from an aerial feature map computed from a local aerial image centered on the predicted pose. The resulting BEV–aerial feature match defines a per-particle log-likelihood for particle-filter updates. In addition, we learn a frame-level uncertainty estimate that adaptively flattens the observation likelihood for unreliable observations, preventing overconfident particle collapse in ambiguous regions. On two real-world off-road datasets, our method achieves *9.7*× lower absolute trajectory error (ATE) on seen routes and *6.6*× lower ATE on unseen routes than a retrieval-based baseline, while remaining robust under partial canopy cover and shadowing. The system runs in real time at *10 Hz* on an NVIDIA Tesla T4, enabling practical robot deployment.**

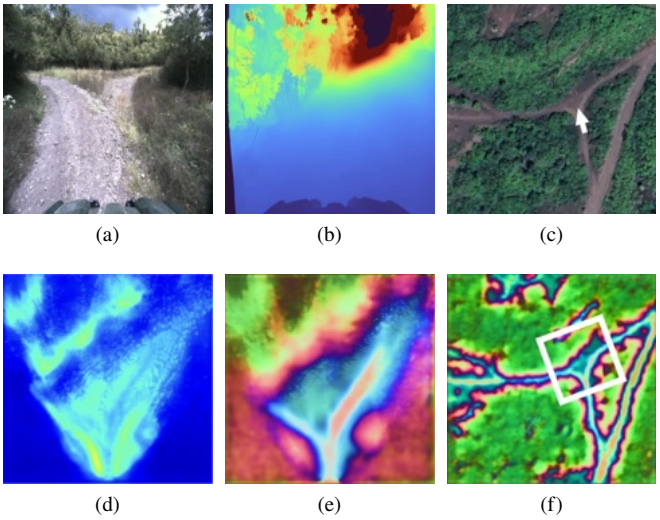


Fig. 1. Visualization of BEV-Patch-PF inputs and outputs. **Top (Inputs):** (a) onboard RGB image $\mathcal{I}$, (b) depth image $\mathcal{D}$, and (c) a local aerial orthophoto $\mathcal{M}[\hat{\mathbf{x}}_{t|t-1}]$, where the white arrow indicates the ground-truth pose. **Bottom (Outputs):** (d) the BEV distinctiveness map **C**, (e) the corresponding feature map **G**, and (f) the aerial feature map **F**. The white box in (f) illustrates one representative sampled patch; during inference, BEV-Patch-PF evaluates a patch for each particle hypothesis.

I. Introduction

High-quality global localization in a geo-referenced frame allows robots to leverage aerial imagery, which can be used to provide improved long-range off-road planning and navigation around hazards such as cliffs and rivers. Although visual and LiDAR-inertial odometry provide local pose estimates, they accumulate drift without global fixes, leading to errors that compromise downstream planning.

Cross-view geo-localization addresses the lack of global position fixes by estimating a robot's 3-DoF pose in a UTM frame by matching ground-level images with geo-referenced aerial imagery. However, this task is inherently difficult due to potentially large viewpoint differences between the onboard and aerial sensors. This problem is especially challenging in unstructured off-road environments, where the absence of man-made landmarks and the presence of terrain irregularities and tree canopy exacerbate the visual mismatch and remove many of the cues that conventional methods rely on [1], [2]. Recent deep learning approaches typically tackle this problem frame-by-frame, falling into two main categories: *retrieval-based* methods [3], [4], [5], [6], [7] that learn global descriptors for ground and aerial images, and *spatial feature-alignment* methods [8], [9], [10], [11], [12] that infer poses by aligning features in a shared representation. Per-frame localization, however, considers only a single observation at a time, making it vulnerable to ambiguity and multi-modal solutions. In off-road settings, this can lead to catastrophic pose jumps caused by visually similar map regions or sensor occlusions. Sequential localization mitigates these issues by enforcing temporal consistency.

While sequential inference can reduce pose ambiguity, it requires observation models that yield smooth, discriminative likelihoods over continuous pose hypotheses. Most prior cross-view methods [5], [6], [8], [9] do not provide continuous likelihoods. Retrieval-based approaches assign similarity scores over a discretized set of aerial patches, making them insensitive to fine-grained pose changes and unsuitable for continuous probabilistic filtering. In contrast,

979-8-3315-8160-2/26/$31.00 ©2026 IEEE

spatial feature-alignment methods offer improved granularity but they either: (i) require dense correlation over discretized pose grids—incurring high computational cost or (ii) optimize a single best pose, which is difficult to use as a likelihood over hypotheses.

To address these limitations, we introduce BEV-PATCH-PF, a sequential localization system that integrates a particle filter with an observation model evaluating likelihoods over continuous pose. From onboard RGB/depth images, we construct a bird's-eye view (BEV) feature map; for each particle pose, we extract the corresponding aerial feature patch and compare it to the BEV features to obtain a per-particle log-likelihood. Because aerial patches can be sampled at arbitrary continuous poses, the likelihood is computed directly at each particle hypothesis, making the approach a natural fit for particle filtering. The model targets unstructured off-road terrain and does not rely on explicit semantic landmarks.

We evaluate our approach on real-world off-road datasets, including TartanDrive [13] and a new dataset called UT-SARA-GQ, which we introduce to specifically test performance under tree canopy. We compare against a retrieval-based pose-graph-optimization method [7] and visual / LiDAR / wheel odometry systems [14], [15]. Across seen and unseen routes from the TartanDrive 2.0 [13] dataset, our method consistently achieves lower trajectory error and greater robustness. These results demonstrate the benefits of modeling continuous-pose likelihoods and confirm generalization to previously unobserved routes.

In summary, our contributions are as follows:

1) A novel observation model for particle filtering that computes continuous-pose likelihoods by matching learned BEV features from ground RGB-D images to features from an aerial orthophoto.
2) Strong performance on off-road localization, with extensive experiments showing significant accuracy gains over existing methods and robust generalization to routes not seen during training.
3) A new public UT-SARA-GQ dataset and benchmark for evaluating cross-view localization under challenging canopy and shadow occlusions, along with experiments validating our method's robustness.
4) A real-time and deployment-ready system, including an open-source C++ ROS 2 wrapper with a TensorRT-optimized inference engine for practical field robotics.

II. Related Works

Visual geo-localization aims to estimate a robot's 3-DoF pose within a geo-referenced map using ground-level imagery. A classic formulation is that of visual place recognition (VPR), where a query image is matched against a pre-collected database of geo-tagged images to find the closest corresponding location [16], [17]. While effective in densely mapped urban areas, VPR is often impractical for off-road missions where comprehensive prior data collection is not feasible.

Cross-view geo-localization with single frames: To overcome the need for a ground-level database, cross-view geo-localization methods match ground images directly to overhead imagery, such as satellite photos or planimetric maps. Early deep-learning approaches focused on learning cross-view descriptors [3], [5], [6], [7]. These methods typically use contrastive learning to align the embedding of a ground image with that of its corresponding aerial patch. However, their accuracy is often limited by the discretization of the aerial map and a lack of explicit orientation modeling. While later work began to infer heading by encoding multiple rotations per grid cell [18], [19], these estimates remain coarse.

To achieve finer pose granularity, spatial-feature-alignment methods were introduced. These techniques, which include dense cross-correlation in BEV space [8], [9] and continuous-pose optimization [10], [11], [12], directly align learned features from both views. Dense correlation methods attain high precision but require sweeping K rotations over an $H \times W$ grid, resulting in a computational complexity of $O(KH^2W^2)$. Conversely, continuous optimization avoids this exhaustive search but is susceptible to converging in local minima.

Sequential estimation for temporal consistency: Per-frame methods are fundamentally challenged by multi-modality and a lack of temporal consistency. Their reliance on single-frame observations provides no mechanism to distinguish between visually similar locations or to ensure the final trajectory is smooth and logical over time. To address this, sequential methods enforce chronological consistency. For instance, OrienterNet [8] warps dense probability maps over time but requires ground-truth odometry. BEVLoc [7] embeds per-frame localizations into a pose graph but requires approximate absolute position fixes (e.g., GPS) to filter outliers. Similarly, a recent end-to-end particle smoother [20] shows strong performance but is confined to urban scenes with planimetric maps.

A common approach for sequential inference is to combine retrieval with a particle filter (PF) [4], [21], [22]. In these systems, each particle represents a pose hypothesis and queries the descriptor of the nearest map cell for comparison with the ground-view descriptor. However, this technique inherits the limitations of retrieval-based methods, namely its dependence on grid discretization and coarse yaw bins, which blurs the likelihood distribution over a continuous pose space.

Off-road cross-view localization: Despite these advances, the problem of off-road cross-view geo-localization remains under-explored. The vast majority of existing methods and datasets focus on structured urban scenes [3], [4], [8], [23]. These environments provide strong structural cues, such as buildings and roads, and often include semantic map annotations that are absent in unstructured terrain. The visual challenges of off-road environments, including texture-poor ground, dense vegetation, and irregular terrain, render the assumptions underlying urban-centric methods untenable. To our knowledge, only BEVLoc [7] and BEVRender [24] have conducted experiments in off-road settings, which suggests that robust localization for off-road environments remains underexplored.

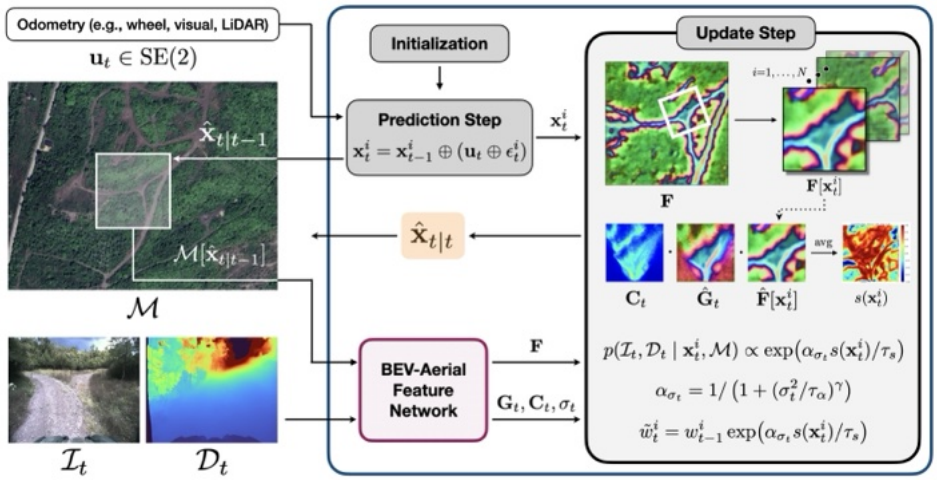


Fig. 2. Overall pipeline of the BEV-Patch-PF. The heatmap in the update step visualizes the weighted similarity used to compute the matching score $s(\mathbf{x}_t^i)$.

III. Particle Filtering with BEV–Aerial Feature Matching

We propose BEV-Patch-PF, a sequential localization framework that combines particle filtering with a learned observation model based on bird's-eye-view (BEV) and aerial feature matching. At each time step, particles are reweighted using a similarity score between the onboard BEV representation and the corresponding aerial feature patch sampled at each particle pose. This recursive filtering loop enables robust localization in challenging environments.

A. Problem Formulation

Our objective is to recursively estimate the ground robot's 3-DoF pose $\mathbf{x}_t = (x_t, y_t, \theta_t) \in \mathrm{SE}(2)$, where (x_t, y_t) are east- and north-directed UTM coordinates (in meters) and θ_t is the heading, measured counter-clockwise from the east axis of a north-up satellite map.

At each timestamp t, the system receives an onboard observation $z_t = \{\mathcal{I}_t, \mathcal{D}_t\}$, consisting of an RGB image and its corresponding depth map, along with a relative motion estimate $\mathbf{u}_t \in \mathrm{SE}(2)$ from an odometry source.

The main problem is to estimate the belief distribution $p(\mathbf{x}_t \mid z_{1:t}, \mathbf{u}_{1:t}, \mathcal{M})$ using a particle filter. The distribution is represented as a set of N weighted samples, $\{(\mathbf{x}_t^i, w_t^i)\}_{i=1}^N$, with each particle $\mathbf{x}_t^i$ denoting a discrete state hypothesis and w_t^i its associated weight. The particle set is updated via Bayesian filtering [25]. Each particle is reweighted according to the likelihood of the current observation z_t:

$$w_t^i \propto w_{t-1}^i \, p(z_t \mid \mathbf{x}_t^i, \mathcal{M}) \tag{1}$$

This reweighting process, paired with a motion prediction step, allows the filter to recursively refine its estimate. Our specific implementation of these steps is detailed next.

B. Particle Filter Localization

The overall BEV-Patch-PF pipeline is illustrated in Fig. 2.

Initialization: Initially, a set of N particles is distributed with Gaussian noise around a coarse initial pose, which can be provided through manual selection. The filter then enters the recursive prediction and update cycle.

Prediction step: Each particle pose $\mathbf{x}_t^i$ at time t is obtained from its predecessor $\mathbf{x}_{t-1}^i$ by propagating the motion estimate $\mathbf{u}_t$ and adding Gaussian noise to account for odometry error:

$$\mathbf{x}_t^i = \mathbf{x}_{t-1}^i \oplus (\mathbf{u}_t \oplus \epsilon_t^i), \quad \epsilon_t^i = \mathrm{Exp}(\boldsymbol{\delta}_t^i). \tag{2}$$

Here, the operator $\oplus$ denotes composition on the SE(2) group, and the process noise ϵ_t^i is generated by sampling a vector $\boldsymbol{\delta}_t^i \in \mathbb{R}^3$ from a zero-mean Gaussian distribution whose covariance is proportional to the odometry $\mathbf{u}_t$, and then mapping it from the Lie algebra $\mathfrak{se}(2)$ to the group SE(2) via the exponential map $\mathrm{Exp}(\cdot)$.

Update step: At each time t, we update the particle weights w_t^i based on the measurement likelihood $p(z_t \mid \mathbf{x}_t^i, \mathcal{M})$.

First, the BEV–aerial feature network (described in Sec. III-C) computes a BEV feature map $\mathbf{G} \in \mathbb{R}^{H_b \times W_b \times D}$, a distinctiveness map $\mathbf{C} \in [0,1]^{H_b \times W_b}$, and a frame-level uncertainty σ_t from the onboard RGB-D image.

Second, for computational efficiency, we avoid processing a separate aerial crop for each of the N particles. Instead, we extract a single larger aerial crop $\mathcal{M}[\hat{\mathbf{x}}_{t|t-1}]$ from the global aerial map $\mathcal{M}$, centered at a representative predicted pose $\hat{\mathbf{x}}_{t|t-1}$ computed from the predicted particle set. This local aerial image is then processed by the network to produce an aerial feature map $\mathbf{F} \in \mathbb{R}^{H_a \times W_a \times D}$.

Third, for each particle hypothesis $\mathbf{x}_t^i$, we sample its corresponding patch $\mathbf{F}[\mathbf{x}_t^i]$ from the aerial feature map $\mathbf{F}$ via bilinear sampling. An affine sampling grid is constructed for each particle, rotated by its heading and anchored at its position, which corresponds to the bottom-center of the patch. This grid is then used to sample a $H_b \times W_b$ patch that is spatially aligned with the BEV feature map $\mathbf{G}$. This approach assumes that the particle distribution is compact enough to remain mostly within the local aerial image $\mathcal{M}[\hat{\mathbf{x}}_{t|t-1}]$. To handle outlier particles that may fall outside this boundary, the sampler uses zero-padding for any out-of-bounds coordinates.

Finally, we compute a matching score $s(\mathbf{x}_t^i) \in [-1, 1]$

for each particle by comparing the BEV feature $\mathbf{G}$ with the sampled aerial feature patch $\mathbf{F}[\mathbf{x}_t^i]$. The distinctiveness map $\mathbf{C}$ assigns higher weights to spatial locations that are more informative for discriminating the correct pose from incorrect pose hypotheses. We compute the score as the mean distinctiveness-weighted cosine similarity over the BEV grid:

$$s(\mathbf{x}_t^i) = \frac{1}{H_b W_b} \sum_{v=1}^{H_b} \sum_{u=1}^{W_b} \mathbf{C}_{uv} \left(\hat{\mathbf{G}}_{uv}^{\top} \hat{\mathbf{F}}[\mathbf{x}_t^i]_{uv} \right). \tag{3}$$

Here, $\hat{\mathbf{G}}$ and $\hat{\mathbf{F}}$ denote the corresponding ℓ_2-normalized BEV and aerial feature maps, respectively.

This matching score $s(\mathbf{x}_t^i)$ is then converted to an observation likelihood, which represents the probability of the current observation given the particle's pose:

$$p(z_t \mid \mathbf{x}_t^i, \mathcal{M}) \propto \exp\big(\alpha_{\sigma_t} s(\mathbf{x}_t^i)/\tau_s\big), \tag{4}$$

$$\alpha_{\sigma_t} = \frac{1}{1 + (\sigma_t^2/\tau_\alpha)^\gamma}. \tag{5}$$

Here, τ_s is a temperature hyperparameter that controls the sharpness of the likelihood distribution, whereas τ_α and γ modulate how the frame-level uncertainty σ_t attenuates the distribution to prevent overconfident updates from uncertain observations. Particle weights are then updated as

$$\tilde{w}_t^i = w_{t-1}^i \exp\big(\alpha_{\sigma_t} s(\mathbf{x}_t^i)/\tau_s\big) \tag{6}$$

and subsequently normalized to obtain w_t^i.

Resampling step: Low-variance resampling is triggered only when the effective sample size falls below a preset threshold, preserving particles most consistent with the true pose while discarding less plausible hypotheses.

C. BEV–Aerial Feature Network

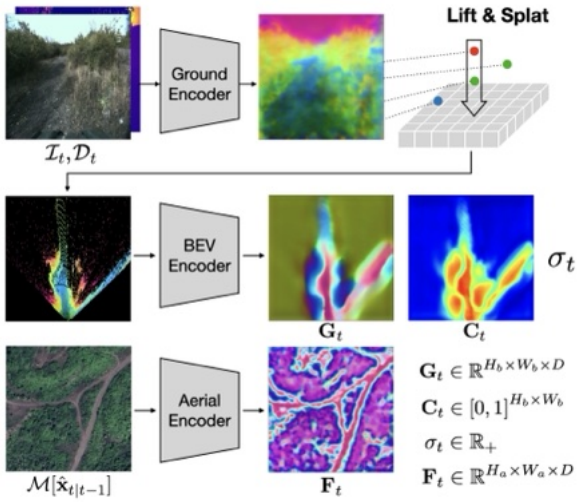


Fig. 3. BEV-Aerial feature network architecture.

The core of our observation model is a feature network that produces the BEV features $\mathbf{G}$, the BEV distinctiveness map $\mathbf{C}$, frame-level uncertainty σ_t, and the local aerial feature map $\mathbf{F}$. As illustrated in Fig. 3, the network contains several modules.

Ground encoder: The ground encoder takes the onboard RGB image $\mathcal{I}_t$ and extracts a feature map. We employ a frozen, pre-trained DINOv3-ConvNeXt-Tiny [26] visual foundation model for its general-purpose feature extraction capabilities. The output features are processed with UPerNet [27] to obtain a higher-resolution feature map, which is then fed to the BEV mapper.

BEV mapper: The 2D features from the ground encoder are projected into a 2D BEV representation by the BEV Mapper. Following the Lift-Splat [28] methodology, we use the depth image $\mathcal{D}_t$ to back-project image features into 3D points in the robot's coordinate frame. To maintain memory efficiency, we avoid creating a dense voxel grid and instead flatten the 3D points into a 2D BEV grid. This grid represents a fixed-size area in front of the robot, defined in its local coordinate frame. The grid's resolution is set to match that of the satellite map. For each BEV grid cell, we compute a height-invariant weighted average of all 3D point features that fall within its vertical column. The weight for each point feature is estimated by a two-layer MLP, allowing the network to prioritize more informative points during splatting.

BEV encoder: The splatted BEV representation is then refined by a BEV Encoder. This module consists of three sequential residual blocks followed by a UPerNet [27] head to aggregate spatial context. The encoder outputs a tensor of shape $\mathbb{R}^{H_b \times W_b \times (D+2)}$. The first D channels form the BEV descriptor map $\mathbf{G}$. The $(D{+}1)$-th channel produces pixel-wise distinctiveness logits, and applying a sigmoid yields the distinctiveness map $\mathbf{C}$. The final channel produces an uncertainty evidence map, whose masked spatial average over valid BEV cells yields a scalar frame-level uncertainty σ_t.

Aerial encoder: The aerial encoder processes the cropped satellite image $\mathcal{M}[\hat{\mathbf{x}}_{t|t-1}]$ using a DINOv3-ConvNeXt-Tiny backbone and a lightweight multi-scale decoder to produce the aerial feature map $\mathbf{F}$. Unlike the ground encoder, which uses a UPerNet head, the aerial encoder uses a substantially lighter decoder that fuses intermediate backbone features with simple projection, upsampling, and shallow convolutional refinement. This design better preserves local pixel-level consistency and empirically reduces blob-like degradation during training.

D. Training Objective

The training objective is designed to learn *(i)* discriminative BEV–aerial matching features, *(ii)* a pixel-wise distinctiveness map for spatial weighting.

Uncertainty-weighted matching loss: To learn a discriminative feature representation, we use an InfoNCE loss over the candidate aerial patches. For each training sample, the ground-truth (GT) pose $\mathbf{x}^+$ serves as the positive, while sampled poses $\mathcal{X}^-$ around the GT pose serve as negatives. We jointly learn a frame-level uncertainty σ and use it to reweight and regularize the matching loss:

$$\mathcal{L}_{\text{match}} = \frac{\mathcal{L}_{\text{sim}}}{\sigma_t^2} + \log \sigma_t^2, \tag{7}$$

$$\mathcal{L}_{\text{sim}} = -\log \frac{\exp(s(\mathbf{x}^+)/\tau)}{\exp(s(\mathbf{x}^+)/\tau) + \sum_{\mathbf{x} \in \mathcal{X}^-} \exp(s(\mathbf{x})/\tau)}. \tag{8}$$

This encourages the model to assign larger uncertainty to difficult observations, such as feature-poor open areas or heavy vegetation. When computing this loss, we stop gradients through the distinctiveness map to prevent the network from trivially increasing distinctiveness weights.

Distinctiveness loss: The distinctiveness map $\mathbf{C}$ is trained in a self-supervised manner to predict which spatial locations are informative for distinguishing the correct pose from incorrect pose hypotheses. Let $\phi_{uv}(\mathbf{x}) = \hat{\mathbf{G}}_{uv}^{\top}\hat{\mathbf{F}}[\mathbf{x}]_{uv}$ denote the per-pixel cosine similarity for pose $\mathbf{x}$. Among the sampled negative poses, we select the hardest negative, $\mathbf{x}_{\text{hard}}^{-} = \arg\max_{\mathbf{x}\in\mathcal{X}^-} s(\mathbf{x})$. The target distinctiveness map is then defined as follows:

$$\begin{aligned}\mathbf{C}_{uv}^{*} = \text{sigmoid}\big(c_p\,\phi_{uv}(\mathbf{x}^{+})\big)\cdot \\ \text{sigmoid}\Big(c_m\left[\phi_{uv}(\mathbf{x}^{+}) - \phi_{uv}(\mathbf{x}_{\text{hard}}^{-})\right]\Big),\end{aligned} \tag{9}$$

where c_p and c_m control the influence of the positive-pose similarity and the similarity margin, respectively. The target $\mathbf{C}^*$ is high at pixels that match the positive patch well while also separating it from the hardest negative patch. A binary cross-entropy (BCE) loss then trains the predicted distinctiveness map $\mathbf{C}$ to match this target:

$$\mathcal{L}_{\text{distinct}} = \text{BCE}\big(\mathbf{C}, \mathbf{C}^{*}\big). \tag{10}$$

The final training objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{match}} + \lambda_{\text{d}}\,\mathcal{L}_{\text{distinct}} \tag{11}$$

IV. Experiments

We evaluate BEV-Patch-PF on offline datasets and real-time onboard deployments to answer four key questions:

- Q1) **Tracking accuracy:** How accurately does our method localize across different platforms and environments?
- Q2) **Generalization:** How robustly does the method perform on routes not seen during training?
- Q3) **Canopy/shadow robustness:** How reliably does the method localize under tree canopy cover and shadowing?
- Q4) **Real-time performance:** Does the system meet onboard compute and latency constraints for real-time operation?

A. Experimental Setup

Datasets: We evaluate on three challenging off-road datasets:

1) TartanDrive 2.0 [13]: Collected with an ATV, this dataset includes 58 trajectories, which we split into 27 for training, 9 for validation, and 22 for testing. The test set is partitioned into 6 seen routes (overlapping training paths) and 16 unseen routes (novel paths).
2) UT-SARA-GQ: A dataset we collected with a Clearpath Warthog in areas with tree-canopy cover and strong shadowing. Totaling 8.3 km and 60k frames, the dataset consists of 15 trajectories, split into 9 for training, 2 for validation, and 4 for testing.
3) Urban park: To evaluate real-time performance, we collected an additional dataset in a local urban park. It contains 5 trajectories, which we split into 2 for training, 1 for validation, and 2 for testing.

Georeferenced imagery: For all experiments, we use north-up RGB satellite orthophotos (GeoTIFFs). To improve model robustness, we train using a dynamically sampled image resolution ranging from 0.15 m/px to 0.45 m/px. For evaluation, we use a fixed resolution of 0.3 m/px. All imagery was reprojected to the appropriate UTM zone using QGIS [29].[1]

Baselines: We compare our method against the following baselines. For offline evaluation, all methods are initialized with the ground-truth starting pose of each trajectory to isolate drift accumulation. Note that BEV-Patch-PF uses no GPS/GNSS or other absolute position fixes during inference.

1) BEVLoc [7]: A recent cross-view localization method. We retrained the official code on our data splits and used stereo visual odometry as its motion prior. Following its original design, BEVLoc periodically applies absolute position fixes to regularize the pose graph; for TartanDrive, we provide the ground-truth positions available in the dataset as oracle position fixes, together with ground-truth orientation for heading assistance.
2) PyCuVSLAM [30]: A high-performance stereo visual odometry baseline.
3) Super Odometry [15]: A LiDAR-Inertial odometry baseline using the pre-computed trajectories provided with TartanDrive 2.0.

Evaluation metrics: We report Absolute Trajectory Error (ATE) in meters, computed as the root-mean-square error (RMSE) between the estimated and ground-truth trajectories in the UTM coordinate frame. No post-alignment is performed.

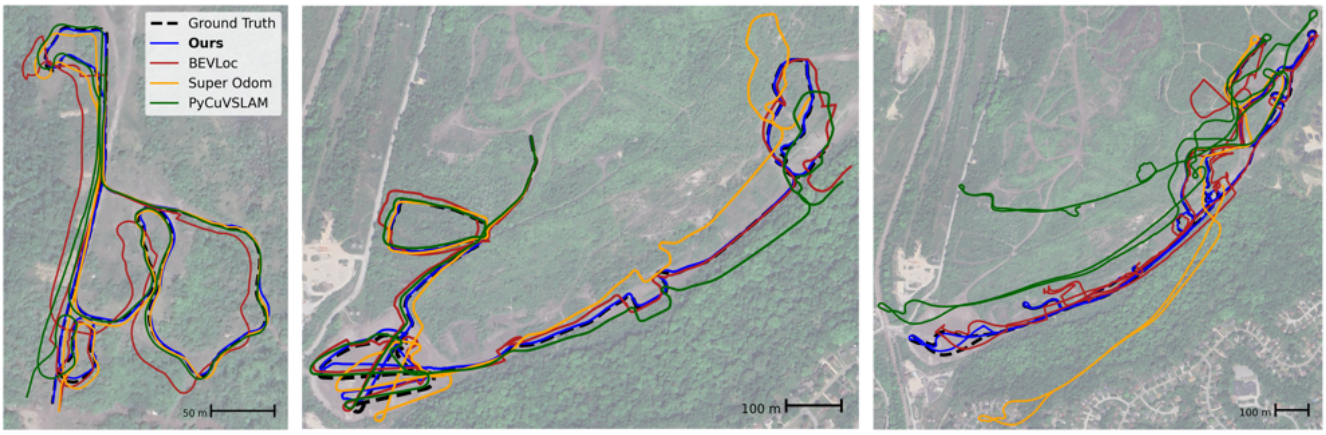
B. Implementation Details

Network architectures: The ground encoder uses a frozen DINOv3-ConvNeXt-Tiny [26] backbone with a UPerNet [27] head to process a 512×512 onboard image. The UPerNet aggregates the $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, and $\frac{1}{32}$ backbone feature maps. The resulting feature map is then processed by the BEV mapper and BEV encoder to produce the final 32-dimensional BEV feature map $\mathbf{G}$, distinctiveness map $\mathbf{C}$, and frame-level uncertainty σ_t. The BEV grid size is 224×224.

The aerial encoder uses the same DINOv3-ConvNeXt-Tiny backbone with a lightweight multi-scale decoder to process a 768×768 aerial image. The decoder takes the $\frac{1}{4}$, $\frac{1}{8}$, and $\frac{1}{16}$ backbone feature maps, projects each scale with a 1×1 convolution, upsamples the coarser scales to the $\frac{1}{4}$ resolution, concatenates them, and applies a shallow convolutional head to produce the final 32-dimensional aerial feature map $\mathbf{F}$.

Training details: We train the network for $25k$ iterations on 4× NVIDIA Quadro RTX 6000 GPUs with a batch size of 4, using data from both the TartanDrive 2.0 and UT-SARA-GQ datasets. We use the AdamW optimizer with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-3}. We use 63 negative poses per positive sample, sampled from translation offsets in the range [5.0m, 100.0m] and heading offsets in the range $[-90°, 90°]$. The InfoNCE temperature parameter τ is learnable, initialized at 0.05 with a minimum value of 0.01.

[1]We reproject Google Satellite imagery to the target UTM zones: 17N (TartanDrive 2.0), 18N (UT-SARA-GQ dataset), and 14N (Urban Park).



(a) Seen route (TD01) (b) Unseen route (TD10) (c) Unseen route (TD22)

Fig. 4. Comparison of estimated trajectories from BEV-Patch-PF and all baselines on the TartanDrive 2.0 dataset.

The distinctiveness coefficients c_p and c_m are set to 10.0 and 2.0, respectively. The loss weights are set to λ_{d}=1.0 for the distinctiveness loss.

Particle filter configuration: For all experiments, we use N = 128 particles, a number chosen to balance tracking accuracy with the computational constraints of onboard deployment. The filter is initialized around the ground-truth starting pose with Gaussian noise (std_{xy}=3.0m, std_{θ}=10°). Prediction noise is proportional to the odometry, with standard deviations set to 10% of the measured motion. The likelihood temperature τ_s is fixed at 1.0. Resampling is triggered when the effective sample size drops below 30%.

C. Q1 & Q2: Accuracy and Generalization

Table I presents the quantitative ATE results, while Figure 4 provides a qualitative comparison of the trajectories. For tracking accuracy, our method achieves an average ATE of 3.10 m, significantly outperforming BEVLoc (21.90 m) and the odometry baselines. Compared with BEVLoc, which often exhibits discontinuities when per-frame ambiguities are not fully resolved by its pose graph, our sequential filtering approach produces smooth and accurate trajectories without using absolute position fixes during inference.

For generalization, on routes not seen during training, BEV-Patch-PF maintains a low ATE of 3.61 m, again surpassing all baselines. This demonstrates that BEV-Patch-PF generalizes well to novel paths while maintaining high accuracy. The cumulative error distribution in Fig. 5 further confirms that our method maintains a substantially tighter error profile on both seen and unseen routes.

D. Q3: Canopy and Shadow Robustness

To test robustness under challenging aerial conditions, we use the UT-SARA-GQ dataset, which contains tree-canopy cover and shadowed trail segments. As shown in Table II and qualitatively in Figure 6, BEV-Patch-PF maintains track lock and estimates accurate trajectories, demonstrating that the learned features remain effective under these real-world appearance variations.

E. Q4: Real-time Performance

We deployed our system on a Clearpath Jackal robot to evaluate its real-time performance. The network was compiled with TensorRT [31] and wrapped in ROS 2, achieving 10 Hz on an NVIDIA Tesla T4 GPU (see Table III for a module-level latency breakdown). During live tests in an Urban Park, the system produced accurate trajectories using only wheel odometry for the motion-prediction step (Table II, Fig. 7). For this deployment, the particle filter was initialized by manually selecting an initial pose in a GUI, demonstrating a fully GPS-free operational workflow.

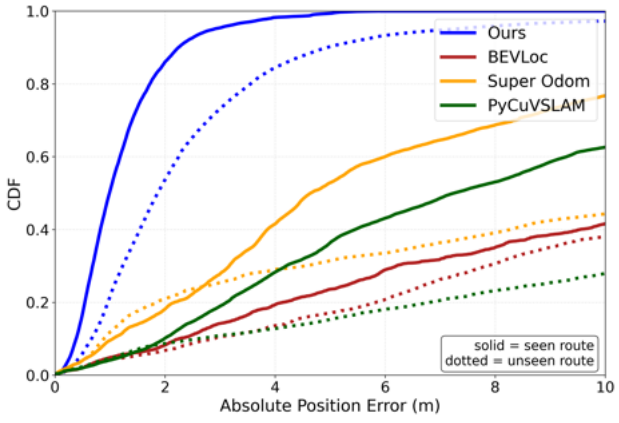


Fig. 5. Cumulative distribution of absolute pose error (meters) for seen and unseen routes on the TartanDrive 2.0 dataset.

V. Conclusion, Limitations, and Future Work

This paper presents BEV-Patch-PF, a sequential cross-view geo-localization system that integrates a particle filter with a learned observation model. By scoring continuous pose hypotheses through BEV–aerial feature matching, BEV-Patch-PF provides accurate vision-only global localization in the UTM frame. Across real-world off-road experiments, the method consistently outperformed odometry and retrieval-

TABLE I. Absolute Trajectory Error (ATE RMSE, meters) on the TartanDrive 2.0 dataset, evaluated in the UTM frame. All methods are initialized with the ground-truth pose. The table reports performance on routes seen during training (TD01–06) and unseen routes (TD07–22). Best results are shown in bold.

	Seen route (6 scenes)						Unseen route (16 scenes)				
Method	TD01	TD02	TD03	TD04	TD05	TD06	TD07	TD08	TD09	TD10	TD11
BEVLoc [7]	16.15	24.78	17.07	33.84	5.97	3.06	23.75	16.63	17.22	22.69	26.30
PyCuVSLAM [30]	8.04	4.61	16.85	12.08	2.49	8.26	38.69	32.87	29.32	35.15	15.08
Super Odometry [15]	5.67	15.50	12.63	3.31	4.76	17.25	8.76	16.12	5.11	54.12	86.38
Ours	**2.10**	**1.62**	**1.11**	**1.64**	**1.57**	**2.38**	**1.61**	**3.00**	**4.02**	**7.64**	**2.24**

	Unseen route (continued)										
Method	TD12	TD13	TD14	TD15	TD16	TD17	TD18	TD19	TD20	TD21	TD22
BEVLoc [7]	18.16	17.72	12.08	33.16	21.44	4.20	27.05	23.88	25.53	35.38	55.64
PyCuVSLAM [30]	270.90	16.49	10.33	6.61	16.74	15.33	42.40	282.43	57.88	32.86	163.02
Super Odometry [15]	344.16	34.91	285.09	7.63	4.26	3.82	727.19	156.77	24.58	14.02	150.55
Ours	**5.89**	**2.72**	**3.30**	**1.93**	**2.33**	**3.17**	**3.85**	**4.69**	**2.74**	**4.23**	**4.37**

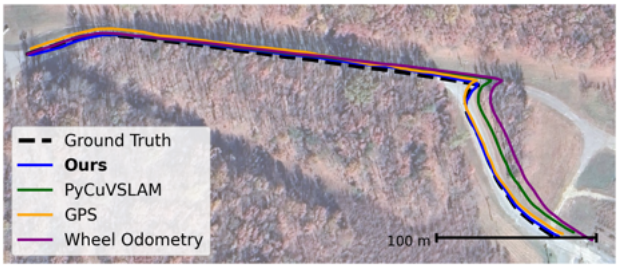


Fig. 6. Example trajectory (GQ01) in the UT-SARA-GQ dataset.

TABLE II. Absolute Trajectory Error (ATE RMSE, meters) on the UT-SARA-GQ dataset and real-time robot deployment at the Urban Park.

	Seen route		Unseen route		Real-time	
Method	GQ01	GQ02	GQ03	GQ04	UP01	UP02
GPS	**3.61**	8.19	4.61	5.07	2.34	3.45
PyCuVSLAM	6.55	13.32	2.79	4.31	-	-
Wheel Odom	11.82	31.04	3.38	7.39	74.89	97.09
Ours	3.68	**7.09**	**2.05**	**4.10**	**2.03**	**2.62**

based baselines, while remaining sufficiently efficient for real-time onboard deployment.

A current limitation is that BEV-Patch-PF performs best within the training distribution, where route semantics, trail geometry, and aerial appearance are similar to those represented in the training data. Performance tends to degrade in out-of-distribution environments, especially when routes are semantically different from the training set, when aerial and ground appearances differ substantially, or when the traversable route is too narrow to be reliably resolved in the aerial imagery. In these cases, the observation model becomes less informative, leading to higher uncertainty and lower localization accuracy.

TABLE III. Per-module inference latency (ms) using TensorRT (FP16).

	Ground Encoder	BEV Mapper	BEV Encoder	Aerial Encoder	Patch Sampler	Scoring Head	Total (+I/O)
RTX 3080	3.85	3.82	1.17	3.96	2.91	5.80	34.96
Tesla T4	10.75	11.64	3.47	11.04	9.84	27.66	92.36

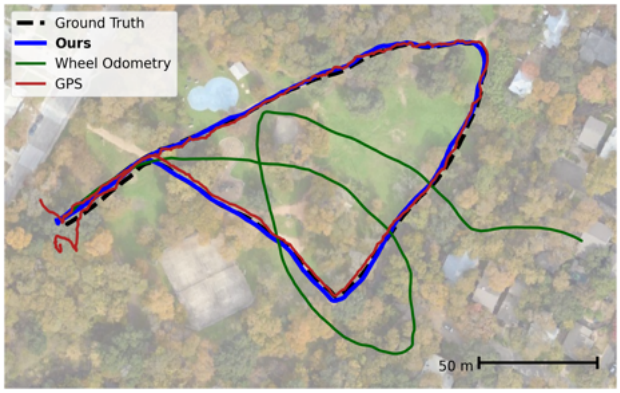


Fig. 7. Trajectory from the real-time experiment (UP02) in the Urban Park. This run was manually initialized in the map GUI.

Future work will focus on improving robustness to out-of-distribution conditions by training on larger and more diverse datasets with wider variation in cameras, capture heights, and environments, such as the Mapillary Street-level Sequences Dataset [32]. We also plan to localize against live drone imagery to mitigate aerial-map mismatch and to incorporate observability-aware path planning that favors routes expected to remain confidently localizable by BEV-Patch-PF.

Acknowledgments

This work is partially supported by the ARL SARA (W911NF-24-2-0025 and W911NF-23-2-0211). Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] F. Castaldo, A. Zamir, R. Angst, F. Palmieri, and S. Savarese, "Semantic cross-view matching," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 9–17.

[2] Y. Tian, C. Chen, and M. Shah, "Cross-view image matching for geo-localization in urban environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3608–3616.

[3] S. Zhu, T. Yang, and C. Chen, "Vigor: Cross-view image geo-localization beyond one-to-one retrieval," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3640–3649.

[4] Z. Xia, O. Booij, M. Manfredi, and J. F. Kooij, "Cross-view matching for vehicle localization by learning geographically local representations," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5921–5928, 2021.

[5] ——, "Visual cross-view metric localization with dense uncertainty estimates," in *European Conference on Computer Vision*. Springer, 2022, pp. 90–106.

[6] S. Zhu, M. Shah, and C. Chen, "Transgeo: Transformer is all you need for cross-view image geo-localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1162–1171.

[7] C. Klammer and M. Kaess, "Bevloc: Cross-view localization and matching via birds-eye-view synthesis," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 5656–5663.

[8] P.-E. Sarlin, D. DeTone, T.-Y. Yang, A. Avetisyan, J. Straub, T. Malisiewicz, S. R. Bulo, R. Newcombe, P. Kontschieder, and V. Balntas, "Orienternet: Visual localization in 2d public maps with neural matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 632–21 642.

[9] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Uncertainty-aware vision-based metric cross-view geolocalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 621–21 631.

[10] Y. Shi and H. Li, "Beyond cross-view image retrieval: Highly accurate vehicle localization using satellite image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 010–17 020.

[11] Z. Song, J. Lu, Y. Shi, *et al.*, "Learning dense flow field for highly-accurate cross-view camera localization," *Advances in Neural Information Processing Systems*, vol. 36, pp. 70 612–70 625, 2023.

[12] Y. Shi, F. Wu, A. Perincherry, A. Vora, and H. Li, "Boosting 3-dof ground-to-satellite camera localization accuracy via geometry-guided cross-view transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 516–21 526.

[13] M. Sivaprakasam, P. Maheshwari, M. G. Castro, S. Triest, M. Nye, S. Willits, A. Saba, W. Wang, and S. Scherer, "Tartandrive 2.0: More modalities and better infrastructure to further self-supervised learning research in off-road driving tasks," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 606–12 606.

[14] NVLabs, "Pycuvslam," https://github.com/NVlabs/PyCuVSLAM, 2025, accessed: 2025-04-25.

[15] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8729–8736.

[16] N. Keetha, A. Mishra, J. Karhade, K. M. Jatavallabhula, S. Scherer, M. Krishna, and S. Garg, "Anyloc: Towards universal visual place recognition," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1286–1293, 2023.

[17] F. Lu, X. Lan, L. Zhang, D. Jiang, Y. Wang, and C. Yuan, "Cricavpr: Cross-image correlation-aware representation learning for visual place recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 772–16 782.

[18] Z. Xia, O. Booij, and J. F. Kooij, "Convolutional cross-view pose estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3813–3831, 2023.

[19] T. Lentsch, Z. Xia, H. Caesar, and J. F. Kooij, "Slicematch: Geometry-guided aggregation for cross-view pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 225–17 234.

[20] A. Younis and E. Sudderth, "Learning to be smooth: An end-to-end differentiable particle smoother," *Advances in Neural Information Processing Systems*, vol. 37, pp. 7125–7155, 2024.

[21] S. Hu and G. H. Lee, "Image-based geo-localization using satellite imagery," *International Journal of Computer Vision*, vol. 128, no. 5, pp. 1205–1219, 2020.

[22] M. Zhou, X. Chen, N. Samano, C. Stachniss, and A. Calway, "Efficient localisation using images and openstreetmaps," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5507–5513.

[23] S. Agarwal, A. Vora, G. Pandey, W. Williams, H. Kourous, and J. McBride, "Ford multi-av seasonal dataset," *The International Journal of Robotics Research*, vol. 39, no. 12, pp. 1367–1376, 2020.

[24] L. Jin, W. Dong, W. Wang, and M. Kaess, "Bevrender: Vision-based cross-view vehicle registration in off-road gnss-denied environment," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 11 032–11 039.

[25] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[26] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, *et al.*, "Dinov3," *arXiv preprint arXiv:2508.10104*, 2025.

[27] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 418–434.

[28] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer, 2020, pp. 194–210.

[29] QGIS Development Team, *QGIS Geographic Information System*, QGIS Association, 2025. [Online]. Available: https://www.qgis.org

[30] A. Korovko, D. Slepichev, A. Efitorov, A. Dzhumamuratova, V. Kuznetsov, H. Rabeti, and J. Biswas, "cuvslam: Cuda accelerated visual odometry," 2025. [Online]. Available: https://arxiv.org/abs/2506.04359

[31] NVIDIA Corporation. (2025) Nvidia tensorrt. [Online]. Available: https://developer.nvidia.com/tensorrt/

[32] F. Warburg, S. Hauberg, M. Lopez-Antequera, P. Gargallo, Y. Kuang, and J. Civera, "Mapillary street-level sequences: A dataset for lifelong place recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2626–2635.