Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Measurement and Information Systems
Department of Telecommunications and Artificial Intelligence

Fusing Point Clouds with RF Signals for Robust Localization

Attila Bóta
author
Budapest University of Technology
and Economics
BSc Computer Engineering
Nokia Bell Labs

Csaba Máté Józsa, Phd
supervisor
Nokia Bell Labs
Artificial Intelligence Research Lab
Senior Research Scientist

Dániel Hadházi
supervisor
Budapest University of Technology
and Economics
Department of Measurement and
Information Systems
Scientific Assistant

2024, Scientific Student Conference

Abstract

Traditional localization systems heavily rely on visual information, which can be adversely affected by lighting conditions, dynamic objects, and repetitive patterns. As single image-based methods lack robustness, we use multiple images to structurally describe the space, thereby modeling the localization problem as a point cloud registration problem. However, achieving robust localization on large maps, such as office buildings and warehouses, poses significant challenges. These environments often exhibit high redundancy and frequent changes in local structure, making point cloud data alone insufficient. Integrating data from additional sensors, such as Wi-Fi or Bluetooth, can substantially improve localization outcomes.

To achieve such robustness, high-quality training data is required. Current datasets suffer from ground truth inaccuracies (derived from Structure-from-Motion or SLAM), lack of large-scale scenes, absence of changes in local structure, and limited sensor diversity, predominantly containing driving data that is often not realistic. To address these issues, we created a comprehensive point cloud registration benchmark, leveraging raw data from the LaMAR dataset. This resource includes recordings from three devices (laser scanner, AR headset, and phone), totaling over 100 hours and covering more than 80,000 square meters over two years.

In order to obtain accurate ground truth poses for the query sessions in the laser-scanned maps, advanced techniques such as image-based localization, pose graph optimization, bundle adjustment, and multi-scale ICP refinement were employed, followed by filtering based on computed inlier ratios. From the query images, we generate bundle-adjusted point clouds using the relative pose between the query images and the corresponding noise-filtered depth maps.

This work has three main contributions. First, we reframe the localization challenge as a point cloud registration problem, leveraging multiple images to structurally describe the space. Second, we introduce a multimodal sensor fusion encoder network that creates transformation-invariant geometric features by fusing 3D point clouds and RF fingerprints. By integrating RF signals, we enhance the ability to disambiguate repetitive patterns and dynamic changes in large-scale environments, resulting in a more robust and reliable localization system. Third, we introduce a hierarchical matching strategy for large point cloud maps. This approach significantly reduces computational complexity by performing coarse-to-fine matching, thereby narrowing down the search space progressively. This hierarchical method ensures that even in extensive and complex environments, the system can maintain high accuracy and performance without overwhelming computational resources.

By addressing the limitations of existing datasets and incorporating advanced data fusion techniques, our work provides improvements in the robustness and accuracy of localization systems. This makes them more resilient to environmental changes and sensor limitations, helping in the construction of more reliable applications in various domains, including industrial metaverse, autonomous navigation, and augmented reality.

Kivonat

A hagyományos lokalizációs rendszerek nagymértékben a vizuális információkra támaszkodnak, amelyek érzékenyek lehetnek a fényviszonyokra, dinamikus objektumokra és ismétlődő mintákra. Mivel az egyképes módszerek nem elég robusztusak, több képet használunk a tér strukturális leírására, így a lokalizációs problémát pontfelhő-regisztrációs problémaként modellezzük. Azonban a robusztus lokalizáció elérése nagy térképeken (például irodaházakban és raktárakban) jelentős mérnöki kihívásnak tekinthető. Ezen környezetek esetén gyakran sok a hasonló objektum, valamint azok sokszor nem statikusak, így a pontfelhő adatok önmagukban nem elegendőek. További szenzorok, például Wi-Fi vagy Bluetooth adatainak integrálása jelentősen javíthatja a lokalizáció pontosságát.

Az ilyen robusztusság eléréséhez kiváló minőségű tanító adatokra van szükség. A jelenlegi adatkészletek pontatlanok a referencia adatokban (Structure-from-Motion vagy SLAM alapján), hiányoznak belőlük a nagyméretű terek, a helyi struktúrák változásai, és az előállításukhoz alkalmazott szenzorok nem változatosak, illetve túlnyomórészt vezetési adatokat tartalmaznak, amelyek gyakran nem realisztikusak. Ezen problémák meghaladása érdekében létrehoztunk egy átfogó pontfelhő regisztrációs benchmarkot, a LaMAR adatkészlet nyers adatait felhasználva. Ez az adatkészlet három eszköz (lézerszkenner, AR headset és telefon) felvételeit tartalmazza, összesen több mint 100 órányi adatot, több mint 80 000 négyzetméternyi területet lefedve, melyet két éven át gyűjtöttek.

A tanítási felvételek lézerszkennelt térképeken való pontos referenciapozícióinak meghatározása érdekében fejlett technikákat alkalmaztunk (például képalapú lokalizáció, pozíció gráf optimalizálás, "bundle adjustment" és többskálás ICP finomítás), majd a számított illeszkedési arányok alapján szűrtük az adatokat. A lekérdezési képekből reprojekció minimalizált (bundle-adjusted) pontfelhőket generáltunk a megfelelő zajszűrt mélységtérképek közötti relatív póz felhasználásával.

A dolgozat három fő szakmai eredményt tartalmaz. Első, hogy újrafogalmazzuk a lokalizációt pontfelhő-regisztrációs problémaként, több képet felhasználva a tér strukturális leírására. Másodszor, bemutatunk egy multimodális szenzorfúzió kódoló hálót, amely transzformáció-invariáns geometriai jellemzőket hoz létre a 3D pontfelhők és RF lenyomatok fúziójával. Az RF jelek integrálásával javítjuk az ismétlődő minták és a dinamikus változások megkülönböztetésének képességét nagyméretű környezetekben, ami robusztusabb és megbízhatóbb lokalizációs rendszert eredményez. Harmadszor, bemutatunk egy hierarchikus illesztési stratégiát nagy pontfelhő térképekhez. Ez a megközelítés jelentősen csökkenti a számítási komplexitást azáltal, hogy coarse-to-fine illesztést végez, így fokozatosan szűkíti a keresési teret. Ez a hierarchikus módszer biztosítja, hogy a rendszer még kiterjedt és összetett környezetekben is magas pontosságot és teljesítményt tudjon fenntartani anélkül, hogy túlterhelné a számítási erőforrásokat.

A meglévő adatkészletek korlátjainak kezelésével és fejlett adatfúziós technikák beépítésével munkánk javíthatja a lokalizációs rendszerek robusztusságát és pontosságát. Ezáltal ellenállóbbá válhatnak a környezeti változásokkal és a szenzorok korlátaival szemben, és megbízhatóbb alkalmazásokat tehetnek lehetővé különböző területeken, beleértve az ipari metaverzumot, az autonóm navigációt és a kiterjesztett valóságot.

Table of Contents

1 Introduction

1.1 Motion and Mapping

Simultaneous Localization and Mapping (SLAM) is a fundamental technique in robotics and autonomous systems that allows an agent to build a map of an unknown environment while simultaneously localizing itself within that map, all in real time. SLAM serves as the cornerstone for enabling machines to navigate and understand their surroundings without prior knowledge. Modern SLAM approaches are increasingly integrating deep neural networks to enhance feature extraction, allowing systems to operate effectively in complex, dynamic environments [53, 39]. Innovations such as multi-modal SLAM, which fuses data from different sensor types (e.g., cameras and IMUs), are extending SLAM's versatility [28].

Odometry is a critical component that underpins SLAM and broader autonomous navigation systems. It provides incremental motion estimation by accumulating small changes in position and orientation using sensor data. This process is essential for tasks like autonomous navigation, robotic exploration, and augmented reality. Sensors such as wheel encoders, inertial measurement units (IMUs), GPS, and cameras (as in visual odometry [29, 13]) are employed to track motion and velocity.

A key challenge in odometry is drift—the gradual accumulation of error in localization as the system moves through the environment. Because odometry relies on incremental updates, small errors in each step can accumulate over time, leading to significant discrepancies between the estimated and actual positions. This drift can compromise the accuracy of the navigation system and the reliability of the generated map.

1.2 Localization

To mitigate drift, methods like **localization** and **re-localization** are employed. Localization involves determining an agent's precise position and orientation within an environment by estimating the transformation between coordinate frames, which encompasses both rotation and translation. This transformation allows the agent to relate its local coordinate system to a global or map coordinate system, enabling accurate navigation and interaction with the environment. Precise localization also enables systems to make informed decisions based on their location, ensuring effective task execution in both dynamic and static settings.

Re-localization enables the system to recognize previously mapped areas and correct its position accordingly, even after significant drift has occurred. Loop closure detection—where the system identifies previously visited locations—also helps recalibrate the map and correct accumulated errors. Visual place recognition plays a critical role in both re-localization and loop closure by leveraging distinctive visual features to match locations, ensuring that even subtle revisits are detected. Advanced filtering and optimization techniques further refine the map's accuracy, ensuring robust localization over long trajectories.

In autonomous vehicles, precise localization is crucial for safe navigation through complex city streets and highways, while drones leverage it for accurate deliveries and inspections in sectors like agriculture and logistics [21]. Moreover, localization is foundational to immersive experiences in the Metaverse, where users interact with virtual objects as if they were real, opening new avenues for entertainment, education, and social engagement [23]. In the Industrial Metaverse, localization transforms industries by enabling precise, real-time collaboration between machines and humans, from smart factories to large-scale construction sites, enhancing efficiency and safety. As these applications evolve, the demand for robust and reliable localization systems becomes increasingly critical, serving as the foundation for the next generation of technological innovation.

1.3 Challenges of robust localization

Despite its widespread use, localization is far from a solved problem. Traditional methods are prone to error, particularly in dynamic environments or areas with poor sensor data. One significant challenge is varying lighting conditions, which can drastically affect the performance of visual sensors. Changes in illumination can lead to inconsistent feature detection and matching, as shadows, reflections, and lighting transitions impact the visual appearance of the environment. This variability makes it difficult for algorithms to reliably recognize and track landmarks, especially in outdoor settings where lighting can change rapidly.

Another complexity arises from different camera types and calibrations. Variations in camera sensors, lenses, and calibration parameters can result in discrepancies in the captured images. These differences affect the accuracy of feature extraction and pose estimation, as the same scene may appear differently to different cameras. Ensuring that localization algorithms are robust to such hardware variations is crucial for widespread applicability across different devices and platforms.

As environments grow more complex and dynamic, the challenge of maintaining real-time, high-precision localization becomes increasingly difficult. Moving objects, changing scenes, and sensor noise further complicate the localization process. Solving these problems could unlock new possibilities in robotics, autonomous navigation, and human-computer interaction, enabling more reliable and versatile systems capable of operating in a wide range of conditions.

2 Related Work

2.1 Visual Feature-Based Localization

Visual feature-based localization operates by utilizing a pre-constructed map of images, each enriched with visual descriptors extracted from distinct features within the images (e.g., edges, corners, or blobs). These descriptors are typically generated using feature detection algorithms like **SIFT** [25] or **ORB** [40]. When a new query image is captured by the agent, the same feature detection and descriptor extraction process is applied. The system then matches the descriptors from the query image to those in the map database, identifying correspondences that can be used to estimate the agent's position and orientation relative to the map. This matching process is fundamental for tasks such as place recognition and loop closure, enabling the agent to recognize previously visited locations and correct for drift.

Bag of Words (BoW) [15] is a foundational method in this domain, which creates a 'visual vocabulary' from the descriptors of the map images. This vocabulary is constructed by clustering similar descriptors into visual words in an offline process, often represented as a hierarchical tree structure. Each map image is then represented as a histogram of these visual words, effectively summarizing its visual content. When processing a query image, its extracted descriptors are quantized using the same visual vocabulary to produce a histogram. The similarity between the query histogram and those of the map images is computed, typically using measures like cosine similarity or Euclidean distance.

To enhance the discriminative power of the matching process, the **term frequency-inverse document frequency (tf-idf)** weighting scheme [45] is employed. The tf-idf score for a visual word in an image is calculated as:

$$\text{tf-idf} = \frac{n_w^i}{n^i} \log\left(\frac{N}{n_w}\right) \tag{1}$$

where:

n_w^i is the number of occurrences of word w in image I^i, n^i is the total number of words in image I^i, N is the total number of images in the map database, n_w is the number of images containing word w. This weighting increases the importance of visual words that are frequent in a particular image but rare across the database, making them more useful for distinguishing between different locations.

The BoW model allows for efficient retrieval of similar images from the map by using an inverted index, which maps visual words to the images containing them. When a query image is processed, the system quickly retrieves candidate images from the map that share common visual words, facilitating real-time localization.

The **Dynamic Bag of Words (DBoW)** extends the traditional BoW approach by allowing the visual vocabulary to be updated incrementally as new images are encountered. This is particularly useful in environments that change over time, such as outdoor scenes with varying lighting conditions or dynamic elements.

In DBoW, both the map images and query images contribute to the evolving visual vocabulary, ensuring that the system remains adaptable to new visual information.

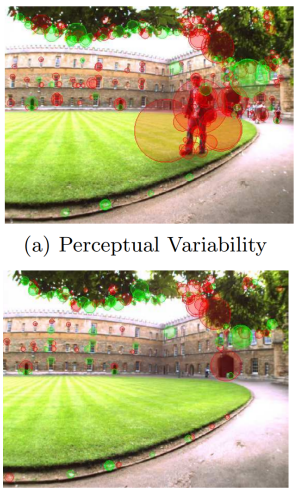
Despite these advancements, visual feature-based localization methods face challenges when dealing with significant variations in viewpoint, illumination, or camera calibrations. For instance, changes in lighting can alter the appearance of features, making descriptor matching less reliable. Different camera types or miscalibrations can introduce distortions that affect the consistency of descriptors between the map and query images.

To address these issues, additional validation steps are incorporated into the localization process. One such method is the two-step geometric validation used in [33], which involves:

1. **Initial Matching**: Performing descriptor matching between the query image and candidate map images to find potential correspondences.

2. **Geometric Verification**: Triangulating the matched features to verify their spatial consistency, ensuring that the correspondences make sense geometrically.

Despite these advancements, visual feature-based localization methods face significant challenges due to environmental variability, sensor differences, and repetitive visual features. Dynamic environments cause two images of the same location to appear different because of changes in lighting, weather, or moving objects like vehicles, complicating feature detection and matching. Variations in camera types, resolutions, and calibrations introduce inconsistencies that can lead to descriptor mismatches between query and map images. Additionally, repetitive patterns such as brickwork, foliage, or common road markings are ubiquitous and can result in ambiguities, as similar descriptors may correspond to different physical locations. These challenges necessitate more robust algorithms capable of handling environmental changes, sensor variations, and feature ambiguities to improve localization reliability.



(a) Perceptual Variability

(b) Perceptual Aliasing

Figure 1: Challenges in visual feature-based localization: (a) Dynamic obstacles, such as a someone walking in front of the camera, introduce changes that complicate feature detection and matching; (b) Repetitive patterns like brickwork can lead to ambiguities in feature matching due to similar visual descriptors appearing in different locations. [5] Words common to both images are shown in green, others in red.

2.2 Learning-Based Localization

In recent years, learning-based localization techniques have transformed visual localization by leveraging deep learning for more accurate, robust, and scalable results. Traditional methods relied on handcrafted features, which often struggle in complex environments with dynamic lighting, changing viewpoints, or repetitive textures. In contrast, deep learning-based methods enhance feature extraction, image matching, and pose estimation, making localization feasible in challenging conditions. These techniques extract robust feature descriptors using convolutional neural networks (CNNs), handle global scene descriptors for large-scale environments, and apply metric learning to improve discriminability. Hierarchical localization frameworks, such as HLoc, exemplify this advancement by combining global descriptors with fine-grained local matching, enabling efficient and reliable localization across a range of environments [41]. This section explores key learning-based techniques and discusses how HLoc integrates these methods to achieve state-of-the-art localization performance.

2.2.1 Deep Feature Extraction and Local Matching

Deep feature extraction and local matching are foundational techniques in learning-based localization, enabling systems to extract and match distinctive features across images. Unlike traditional methods, which rely on handcrafted features, deep learning-based methods learn to detect and describe features that are robust to variations in lighting, viewpoint, and scale.

In this domain, **SuperPoint** [9], **R2D2** [37], and **D2Net** [10] are commonly employed. These methods use convolutional neural networks (CNNs) to identify keypoints and generate feature descriptors, which significantly improve localization performance in challenging environments. SuperPoint operates in a self-supervised framework to produce repeatable and accurate features that adapt well to a variety of scenes. Meanwhile, R2D2 optimizes for both repeatability and distinctiveness, ensuring that its features are well-suited for matching under challenging conditions. D2Net is designed for robustness to scale and rotation, providing accurate keypoints and descriptors for scenes with large perspective changes.

By leveraging these learned features, visual localization pipelines can establish reliable correspondences between images. As these methods evolve, deep feature extraction and local matching continue to play a crucial role in the development of scalable and accurate localization systems.

2.2.2 Global Descriptor Techniques for Place Recognition

Global descriptor techniques play a critical role in place recognition and loop closure detection for visual localization. These methods generate compact representations, or "global descriptors," of an entire image, which allows efficient retrieval of similar images from a large database. Global descriptors enable localization systems to identify previously visited places by matching the global representation of a new image to stored representations, significantly reducing the need for exhaustive pairwise matching.

NetVLAD [1] is a popular global descriptor method that aggregates local features into a single descriptor. Inspired by the classic VLAD (Vector of Locally Aggregated Descriptors) technique, NetVLAD enhances it with a neural network framework, allowing it to learn feature aggregation in an end-to-end manner. This enables it to create robust global descriptors suitable for large-scale environments, making it particularly useful in urban and structured settings where visual repetition is common.

Another widely-used global descriptor technique is **AP-Gem** [16, 36], which uses a differentiable pooling layer, capturing fine-grained visual details by learning adaptive weights for feature aggregation. AP-Gem enables effective place recognition by adapting to varied appearance changes across scenes, and it has demonstrated robust performance in challenging scenarios with significant lighting and viewpoint variations.

By integrating these global descriptors, hierarchical localization frameworks can efficiently narrow down the search space in a database before applying more computationally-intensive local feature matching techniques.

2.2.3 Metric Learning for Feature Discriminability

Metric learning is a core component in learning-based localization that aims to enhance the discriminability of extracted features by learning an embedding space where similar features are closer together and dissimilar ones are pushed apart. In localization, this approach is especially useful for robust feature matching, as it minimizes ambiguity between features from different locations while maximizing similarity for features from the same location.

To achieve this, metric learning often relies on loss functions like the **Triplet Loss** [44], which is designed to optimize the distances among an anchor sample x^a, a positive sample x^p, and a negative sample x^n. The triplet loss encourages the anchor to be closer to the positive than to the negative by a margin α:

$$\mathcal{L}_{\text{triplet}} = \max\left(0, \|f(x^a) - f(x^p)\|^2 - \|f(x^a) - f(x^n)\|^2 + \alpha\right), \tag{2}$$

where $f(x)$ represents the embedding of input x. This loss function ensures that positive pairs are closer in the feature space than negative pairs by at least the margin α.

Alternatively, the **Contrastive Loss** [19] can be employed to separate positive and negative pairs more directly. Given a pair of samples (x_i, x_j) with label $y = 1$ if they are similar and $y = 0$ if dissimilar, the contrastive loss is formulated as:

$$\mathcal{L}_{\text{contrastive}} = y \cdot \|f(x_i) - f(x_j)\|^2 + (1 - y) \cdot \max\left(0, m - \|f(x_i) - f(x_j)\|\right)^2, \tag{3}$$

where m is a margin that enforces separation between dissimilar pairs.

While effective, these loss functions often require careful selection of pairs or triplets and can suffer from slow convergence due to inefficient use of the training data. To address these limitations, the **Circle Loss** [47] has been proposed, offering better gradient properties and more efficient optimization.

Circle loss unifies the advantages of cross-entropy and triplet losses by considering all similarity scores in a unified framework. It assigns adaptive weights to each pair based on their similarity scores, focusing the learning process on more informative examples. The loss is formulated as:

$$\mathcal{L}_{\text{circle}} = -\log \frac{\sum\limits_{(i,j)\in\mathcal{P}} e^{\gamma\cdot\left(s_p^{ij}-\delta_p\right)}}{\sum\limits_{(i,j)\in\mathcal{P}} e^{\gamma\cdot\left(s_p^{ij}-\delta_p\right)} + \sum\limits_{(i,k)\in\mathcal{N}} e^{\gamma\cdot(s_n^{ik}-\delta_n)}}, \tag{4}$$

where:

- $s_p^{ij} = f(x_i)^\top f(x_j)$ is the similarity between positive pairs,
- $s_n^{ik} = f(x_i)^\top f(x_k)$ is the similarity between negative pairs,
- $\mathcal{P}$ and $\mathcal{N}$ denote the sets of positive and negative pairs, respectively,
- γ is a scaling factor,
- δ_p and δ_n are adjustable margins for positive and negative pairs.

Circle loss introduces **angular margins** and **adaptive weighting** for each pair:

$$\alpha_p^{ij} = \left[s_p^{ij} - \delta_p\right]_+, \tag{5}$$
$$\alpha_n^{ik} = \left[s_n^{ik} - \delta_n\right]_+, \tag{6}$$

where $[\cdot]_+$ denotes the positive part, meaning $[x]_+ = \max(x, 0)$. The gradients of circle loss are proportional to these adaptive weights, emphasizing harder samples (i.e., positive pairs with low similarity and negative pairs with high similarity).

The key benefits of circle loss include:

- **Better Gradients**: By providing larger gradients to harder samples, circle loss accelerates convergence and enhances the discriminative power of the embeddings.
- **Unified Optimization**: It simultaneously considers all positive and negative pairs within a mini-batch, eliminating the need for complex sample mining strategies.
- **Adaptive Margins**: The use of adjustable margins allows for more flexible control over the decision boundary between classes.

Through metric learning with circle loss, feature representations become more robust to variations in viewpoint and appearance, improving the reliability of feature matching across diverse and dynamic environments. Circle loss has gained traction in modern localization frameworks due to its ability to create highly discriminative feature spaces with efficient optimization.

2.2.4 Hierarchical Localization

Hierarchical localization combines global and local localization techniques to address scalability and robustness in large-scale environments. The approach divides the localization process into two main stages: an initial coarse localization using global descriptors, followed by fine localization through feature matching. This multi-stage process narrows down potential locations for matching, which reduces computational costs while maintaining high precision.

In the first stage, global descriptor methods like NetVLAD and AP-GeM enable fast retrieval of candidate locations, efficiently reducing the search space. Once candidate locations are identified, a local matching step using deep features such as SuperPoint or D2-Net aligns the query image precisely to the map, enhancing accuracy. This hierarchical approach has become widely used in visual localization and SLAM due to its effective balance between computational efficiency and precision.

HLoc [41] is a prominent framework implementing hierarchical localization, leveraging these global descriptors for initial place recognition and then refining localization through local feature matching. Its flexibility and robust performance have made it popular for large-scale localization tasks.

2.2.5 End-to-End Learning Approaches

End-to-end learning has become increasingly prominent in visual localization due to its ability to jointly optimize feature extraction, matching, and pose estimation within a single network. Unlike traditional approaches with separated stages, end-to-end methods integrate these components into a unified pipeline, reducing dependency on handcrafted features and improving adaptability in diverse environments.

Within the HLoc framework, models such as **HF-Net** [41] offer a fully differentiable architecture, optimizing all stages via backpropagation. HF-Net's integrated approach combines global and local feature extraction, handling both place recognition and precise localization through end-to-end learning. This unified structure not only optimizes each module individually but also allows submodules to learn representations that are beneficial to subsequent stages in the pipeline. For example, features learned during the extraction phase are tailored to improve downstream matching and pose estimation tasks, as they capture details most relevant to accurate localization.

By learning task-specific features directly from data, these models can reduce overfitting to specific visual characteristics or environmental conditions, thus generalizing better across varied settings. Additionally, by mitigating the limitations of handcrafted features, which often struggle with lighting or viewpoint changes, they can learn discriminative global and local descriptors, balancing robustness to visual variations with the precision required for effective place recognition and fine localization.

2.2.6 Challenges of Learning-Based Techniques

While learning-based localization techniques offer enhanced robustness and adaptability, they also come with significant challenges. A primary issue is the high computational demand for both training and inference, as deep neural networks require extensive resources, which can hinder deployment on mobile or embedded platforms. Additionally, achieving generalization across varied scenes remains difficult, as model performance may degrade in unfamiliar environments with novel visual conditions. This is further complicated by the need for large, labeled datasets, which are costly and time-consuming to collect and may introduce biases. These limitations underscore the need for advances in efficiency and scalability to make learning-based localization more viable for a broader range of applications.

2.3 Hypothesis-Based Localization

Hypothesis-based localization methods rely on generating a set of potential hypotheses for an agent's position and iteratively refining these estimates based on sensor data. This probabilistic framework is particularly valuable in environments where deterministic methods may fail, such as in dynamic settings with ambiguous visual features or frequent changes. One of the foundational techniques in this category is Particle Filtering, also known as Monte Carlo Localization [49, 14].

Particle Filtering [17] operates by representing possible positions as particles, each representing a unique hypothesis about the agent's location and orientation. These particles are updated at each time step, weighted according to the likelihood of their position given the sensor measurements. The filter then resamples particles, concentrating them in high-likelihood areas and refining the agent's estimated location. This sequential Monte Carlo approach is robust to noise and can handle complex distributions, making it suitable for non-Gaussian and multimodal environments [8].

However, maintaining and updating a large number of particles is computationally demanding, which can pose challenges in real-time applications or on resource-constrained devices. As a result, various optimizations have been proposed, such as adaptive particle filtering and selective resampling, to balance computational efficiency with localization accuracy [18].

Building upon traditional particle filtering, advanced techniques like hypothesis density filters have been introduced to enhance robustness in dynamic environments. Methods like the Rao-Blackwellized Probability Hypothesis Density (RB-PHD) filter and Single Cluster Probability Hypothesis Density (SC-PHD) filter manage multiple hypotheses efficiently, making them well-suited for localization in scenarios with numerous moving elements or rapid changes in visual appearance [24, 26, 52].

A notable extension of these concepts is GEM-SLAM [11], an optimized self-localization approach for dynamic scenes that leverages hypothesis density filtering. GEM-SLAM integrates density estimation with particle resampling to improve localization precision in challenging environments, achieving substantial performance gains over traditional methods such as RB-PHD and FastSLAM [27]. By handling both static and dynamic elements effectively, GEM-SLAM represents an advanced application of hypothesis-based localization, pushing the boundaries of particle filtering in real-world autonomous navigation.

2.4 Point Cloud Registration

Point cloud registration is a foundational process in 3D computer vision and robotics that aligns multiple 3D point clouds from different viewpoints into a unified coordinate system. This process is essential for applications such as mapping and object recognition, as it enables accurate reconstruction of observed scenes.

2.4.1 Iterative Closest Point

The Iterative Closest Point (ICP) algorithm is a classic approach to point cloud registration, known for its simplicity but limited by slow processing times in complex scenarios. ICP works by iteratively refining an initial transformation to minimize the distance between corresponding points in the source and target clouds. Each iteration includes two steps: finding nearest neighbor correspondences between the source and target point clouds, and computing an optimal transformation to align these points.

The transformation matrix $\mathbf{T}$ is computed by minimizing the mean squared error (MSE) between corresponding points, represented mathematically as:

$$\mathbf{T} = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i} \|\mathbf{R}p_i + \mathbf{t} - q_i\|^2 \tag{7}$$

where p_i are points in the source cloud, q_i are the nearest corresponding points in the target cloud, R is the rotation matrix, and t is the translation vector. Although ICP is effective for static and controlled scenes, it struggles with computational efficiency and is prone to errors in dynamic or noisy environments, particularly when faced with partial data or significant transformations.

For ICP to converge accurately and efficiently, it requires a relatively good initial transformation as a starting point. Without a reasonable initial alignment, the algorithm may converge to a local minimum or take a significantly longer time to reach an accurate solution. This reliance on a good initial guess makes ICP less suitable for applications where precise initial positioning cannot be guaranteed.

2.4.2 GeoTransformer

GeoTransformer introduces a more robust and adaptable approach to point cloud registration by leveraging a geometric transformer-based architecture [34]. This method enhances the registration process by learning invariant features based on local geometric structures and global context, providing resilience to viewpoint changes, scale variations, and partial occlusions.

GeoTransformer improves upon traditional methods by learning robust features; the attention mechanisms enable the capture of long-range dependencies and complex geometric relationships. It effectively handles partial overlaps, allowing the model to register point clouds even when there is only partial overlap. Additionally, by focusing on salient features, the network enhances noise robustness, making it less affected by noise and outliers.

However, the computational complexity of transformer models can lead to high resource consumption in terms of memory and processing power. This can limit the applicability of GeoTransformer in real-time or resource-constrained applications.

2.4.3 BiEquiformer

BiEquiformer pushes the boundaries of point cloud registration by introducing a bidirectional equivariant transformer framework [30]. By integrating the principles of equivariance into transformer architectures, BiEquiformer ensures that learned features remain consistent under geometric transformations like rotations and translations, greatly enhancing robustness.

A key innovation of BiEquiformer is its ability to process source and target point clouds simultaneously in a bidirectional manner, refining features in both directions to improve correspondence accuracy. This bidirectional refinement enhances mutual information between point clouds, enabling the network to resolve ambiguities and preserve minute structural details essential for precise alignment.

Mathematically, the equivariance property in BiEquiformer is defined as:

$$f(g \cdot x) = \rho(g) f(x), \tag{8}$$

where g is a transformation in group G, f is the feature extractor, x is the input point cloud, $\rho(g)$ is the representation of G in the feature space, and $g \cdot x$ denotes the action of g on x.

By leveraging equivariant feature learning and efficient attention mechanisms, BiEquiformer excels at aligning point clouds across various scales and orientations, even in challenging scenarios with low-scale scenes or small objects. Its high efficiency and precision make it an exciting advancement for applications requiring exceptional accuracy, such as object reconstruction, robotics, and fine-grained localization tasks.

2.4.4 Considerations for Practical Deployment

While advanced models like GeoTransformer and BiEquiformer significantly improve point cloud registration performance, considerations regarding computational resources and real-time applicability remain important factors when deploying these models in practical systems. The inclusion of point clouds adds considerable computational overhead and memory requirements, as handling and processing high-density 3D data in real-time applications remains computationally heavy. Their high computational complexity and resource demands can pose challenges for real-time applications or systems with limited processing capabilities. Optimizing these models for efficiency or developing lightweight alternatives is essential for practical deployment in scenarios where computational resources are constrained.

3 Localization as a Point Cloud Registration Problem

In this section we provide more details about the main results of this Thesis.

3.1 Point Cloud Alignment for Localization

In this work, we propose a novel approach to solving the loop closure, re-localization, and co-localization challenges within a SLAM pipeline by treating localization as a point cloud registration problem. Unlike traditional image-based localization methods that rely on 2D features from single images and are often susceptible to variations in lighting and camera inconsistencies, our approach accumulates spatial information over sequences of images to create a detailed 3D point cloud representation of the environment. Alternatively, point clouds can be generated using depth cameras or Lidar, adding flexibility to this method.

We formulate the localization problem as a point cloud registration problem in the following way. Given a global point cloud map $\mathcal{P}$—a large, detailed representation of the environment recorded offline—and a local point cloud $\mathcal{Q}$ captured in real-time by a query device, our objective is to determine a rigid transformation ${}_{\mathcal{P}}\mathbf{T}_{\mathcal{Q}} \in SE(3)$ that aligns $\mathcal{Q}$ within the coordinate system of $\mathcal{P}$. This transformation, denoted as ${}_{\mathcal{P}}\mathbf{T}_{\mathcal{Q}} = \{\mathbf{R}, \mathbf{t}\}$, where $\mathbf{R} \in SO(3)$ is the rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is the translation vector, encapsulates the rotation and translation needed to bring the local map $\mathcal{Q}$ into alignment with the global map $\mathcal{P}$. To achieve this alignment, we calculate correspondences between points in $\mathcal{P}$ and $\mathcal{Q}$. These correspondences are pairs of points that represent similar features or structural elements within the two point clouds, allowing us to establish a direct relationship between the two coordinate frames. By selecting a transformation ${}_{\mathcal{P}}\mathbf{T}_{\mathcal{Q}}$ that best aligns these corresponding points, we can optimize the alignment between $\mathcal{P}$ and $\mathcal{Q}$ with high accuracy. Global point clouds are created by scanning the target environment by means of cameras or Lidars and processed by offline SLAM algorithms with the purpose of creating high-accuracy structural maps of the environment. On the contrary local maps are created online based on smaller data-sequences often relying on Visual-Inertial Odometry (VIO) algorithms.

The core advantage of this approach is its reliance on structural information within the 3D environment to create highly discriminative spatial matching descriptors. Through multiple layers of compression and feature transformation, the model produces robust descriptors capable of generalizing across diverse environmental conditions. By obtaining precise correspondences between point clouds, we can compute the rigid transformations necessary to align them accurately, even in visually challenging environments with limited or repetitive features. This structural reliance provides robustness against variations in lighting and is inherently camera-agnostic, as it transforms visual sequences into 3D representations that are unaffected by camera-specific characteristics like lens distortions and model variations. This eliminates the need for image correction processes or extensive calibration across different camera types, enabling robust localization across diverse devices and environmental conditions.

Consequently, our point cloud-based localization framework achieves a high-level of generalization, making it suitable for reliable localization regardless of lighting, seasonal changes, or equipment differences.

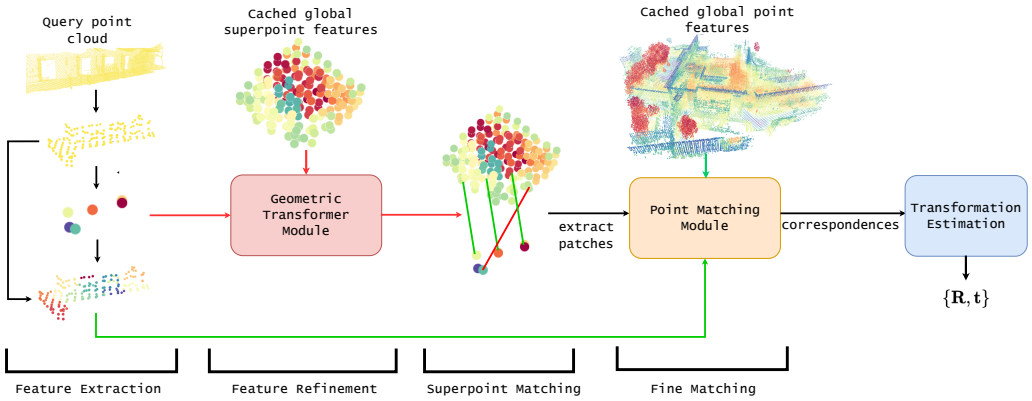
However, working with point clouds also introduces specific challenges related to point cloud-specific noise. Point clouds can suffer from sensor inaccuracies, resulting in measurement noise that introduces errors in the captured 3D data. Issues such as occlusions can lead to missing data, creating incomplete representations of the environment. Additionally, variations in point density due to changes in sensor distance or angle can affect the uniformity of the point cloud, complicating the registration process. These challenges necessitate robust algorithms capable of handling noisy and incomplete data to ensure accurate alignment and reliable localization.

3.2 Caching the Global Map Features for Real-Time Inference

To enable real-time inference, we cache pre-computed features of the global point cloud map. This allows us to retain the most computationally expensive aspect of processing—feature extraction—only once, and reuse these features during inference. During localization, we first match the query device's local point cloud descriptors to the cached coarse feature map of the global point cloud. Once coarse matches are identified, we load only the corresponding finer-level representations for these matched areas and continue matching at progressively finer levels, focusing on the specific patches identified in the previous stage.

By using this strategy, we significantly cut down on the memory requirements, as only the relevant parts of the global map are actively used during alignment. This hierarchical approach eliminates the need to load or process the entire point cloud map at once, progressively narrowing down the search to relevant sections. By refining the matching process layer by layer, we continue until we reach the finest level, where we compute the final transformation or pose based on these refined matches. This enables real-time localization even in large and complex environments, without the prohibitive overhead of processing the entire point cloud.

Figure 2: Architecture During Inference



3.3 Increasing Efficiency with Trajectory Extension

Our approach leverages a hierarchical registration framework that enhances computational efficiency by adopting a trajectory-increasing approach, avoiding the traditional particle filter's reliance on multiple hypotheses. In contrast to particle filtering—where multiple local hypotheses are tested simultaneously, often leading to redundant computations in visually repetitive or feature-scarce environments—our method maintains a single, evolving hypothesis. This hypothesis is progressively refined by capturing additional structure in the environment if initial localization attempts are unsuccessful.

With this trajectory-increasing strategy, we first attempt localization on a coarse level. If confident matches are established at this level, we proceed to finer levels to improve accuracy. This progressive approach enables iterative refinement without redundant computations, as additional data is only captured when necessary. Moreover, because we compute fine features only for the relevant matched regions, this method significantly reduces the computational load compared to traditional exhaustive search methods.

In cases where localization succeeds on a particular density level, the framework allows us to skip unnecessary calculations for previously confirmed coarse patches, further optimizing performance. By caching and reusing already-computed fine features, the system avoids recomputation, needing only to process newly captured data or areas where localization has not yet been confirmed. This layered approach enables us to save substantial computational resources, ensuring efficient alignment even in complex, large-scale environments.

In cases where localization succeeds on a particular density level, the framework allows us to skip unnecessary calculations for previously confirmed coarse patches, further optimizing performance. By caching and reusing already-computed fine features, the system avoids recomputation, needing only to process newly captured data or areas where localization has not yet been confirmed. Specifically, we only need to recompute a feature if the matching isn't confirmed on its density level or if new information becomes available within its receptive field. This layered approach enables us to save substantial computational resources, ensuring efficient alignment even in complex, large-scale environments.

4 Point Registration with Deep Neural Networks

Given two point clouds $\mathcal{P} = \{p_i \in \mathbb{R}^3 \mid i = 1, \ldots, N\}$ and $\mathcal{Q} = \{q_j \in \mathbb{R}^3 \mid j = 1, \ldots, M\}$, our objective is to estimate a rigid transformation $\mathbf{T} = \{\mathbf{R}, \mathbf{t}\}$ that best aligns these two point clouds. Here, $\mathbf{R} \in SO(3)$ represents a 3D rotation matrix, and $\mathbf{t} \in \mathbb{R}^3$ denotes a 3D translation vector.

To determine this transformation, we solve the optimization problem [2]:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{(p^*_{x_i}, q^*_{y_i}) \in \mathcal{C}^*} \left\| \mathbf{R} \cdot p^*_{x_i} + \mathbf{t} - q^*_{y_i} \right\|_2^2, \tag{9}$$

where $\mathcal{C}^*$ is the set of ground-truth correspondences between $\mathcal{P}$ and $\mathcal{Q}$. Since $\mathcal{C}^*$ is typically unknown in practice, the task involves first identifying correspondences between points in $\mathcal{P}$ and $\mathcal{Q}$, followed by estimating the alignment transformation based on these correspondences [20, 34, 30].

Our model architecture addresses the challenges unique to point clouds, such as their unordered structure and varying point densities. To handle these properties effectively, the architecture uses a hierarchical, superpoint-based design that enables learning across multiple scales.

The process begins with grid-based subsampling, which reduces the number of points by grouping them into voxel grids and representing each voxel with an averaged superpoint. This subsampling step reduces computational demands while preserving the spatial structure of the data, forming a foundational representation for further processing.

The architecture then leverages point cloud convolution methods, particularly deformable kernel point convolutions to learn robust, transformation-invariant representations. These convolutions adapt their kernel points to fit the local geometry of the point cloud, enhancing the model's ability to capture fine-grained structural details.

Additionally, self-attention and cross-attention mechanisms refine these representations by incorporating spatial relationships both within and across point clouds, improving the model's accuracy in identifying correspondences, even under challenging conditions.

The model follows a multi-scale structure that progressively refines the matching process. Initial correspondences are established at the coarse superpoint level, followed by identification of finer correspondences within each superpoint's local neighborhood. This approach enhances the accuracy of correspondence matching, leading to a highly precise transformation estimation between the point clouds.

To estimate the final transformation between the point clouds, we solve the alignment problem as shown in Equation (9). A common approach is to use a robust estimator like RANSAC [12], which iteratively refines the transformation by selecting inliers among correspondences. However, given the reliability of our hierarchical correspondences, we can also apply a weighted least-squares approach, leveraging correspondence weights from the hierarchical matching process. This method enhances robustness and computational efficiency, yielding a precise transformation estimate with reduced computation time [4].

4.1 Handling Point Clouds in Deep Learning

Point clouds present unique challenges for deep learning models, primarily due to their unordered nature, differing densities, and lack of inherent structure compared to grid-based data. Efficiently processing and learning from point clouds requires specialized techniques to handle these characteristics. Superpoint-based techniques allow the segmentation of point clouds into meaningful regions, providing a higher-level abstraction that improves processing efficiency. This segmentation is particularly crucial for handling unordered data and diverse density distributions, especially in complex environments with varied structural features [32].

Our approach begins with subsampling, an essential step for building a hierarchical representation of point clouds. By reducing the number of points, subsampling decreases computational load and memory usage, making it feasible to process large-scale point clouds. Moreover, subsampling supports multi-scale feature learning across spatial scales, allowing the model to capture both fine and coarse structural patterns within the data. The coarse levels retain broader spatial context through larger receptive fields, capturing a global overview, while the finer levels offer local refinement, enhancing spatial detail in the representation.

To fully leverage this hierarchical representation, we need connections between levels. These connections allow information to flow both upwards and downwards, integrating coarse-level global context with fine-grained details at deeper levels, enhancing the model's ability to recognize structural relationships and reduce ambiguity. Furthermore, when two superpoints are matched, the model can refine the matching at finer levels by matching their respective patches. Here, a patch refers to the local region around each superpoint, containing the points within its receptive field. This enables the model to confirm and refine alignment progressively, ensuring that matches established at coarser levels are validated and enhanced with higher spatial resolution at finer levels.

In our model, we use a grid-based subsampling method to select superpoints, which serve as the basis for subsampling, upsampling, and inner-layer operations. This subsampling method divides the point cloud into voxel grids of a specified size and computes the average position of points within each voxel, rather than simply selecting the center. This averaging approach captures a more representative location within each voxel, preserving finer details in the distribution of points, especially in regions with uneven densities.

Given an input point cloud $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{N}$, where each point $\mathbf{p}_i \in \mathbb{R}^3$, we define the set of superpoints $\hat{\mathcal{P}}$ by dividing the point cloud space into voxel grids with a specified voxel size v (the edge length of each voxel in the grid). For each voxel, the representative superpoint is computed as the average of all points within that voxel. The location of each superpoint $\hat{\mathbf{p}}_j \in \hat{\mathcal{P}}$ can be expressed as:

$$\hat{\mathbf{p}}_j = \frac{1}{|V_j|} \sum_{\mathbf{p}_i \in V_j} \mathbf{p}_i \tag{10}$$

where V_j denotes the set of points within the j-th voxel, and $|V_j|$ is the number of points in that voxel. This approach ensures that the selected superpoints provide a spatially balanced representation of the point cloud while reducing its overall density.

To facilitate efficient subsampling, upsampling, and inner layer convolution operations, we rely on nearest-neighbor calculations across these different layers. In each case, finding the nearest neighbors is essential for accurately propagating features, especially given the unordered and high-dimensional nature of point cloud data. To achieve efficient neighbor searching, we employ a k-d tree data structure, which allows for rapid nearest-neighbor queries. This is crucial for scalability, as k-d trees significantly reduce the computational cost of finding neighbors within large point clouds.

For upsampling, we use a nearest-neighbor approach, where each finer point $\mathbf{p}_i \in P$ inherits the features of the nearest superpoint $\hat{\mathbf{p}}_j \in \hat{\mathcal{P}}$ based on k-d tree calculations. The nearest-neighbor upsampling can be represented as:

$$f(\mathbf{p}_i) = \hat{\mathbf{p}}_{j^*} \tag{11}$$

where $j^* = \arg\min_j \|\mathbf{p}_i - \hat{\mathbf{p}}_j\|$, meaning $\hat{\mathbf{p}}_{j^*}$ is the nearest superpoint to $\mathbf{p}_i$. This direct transfer of features ensures spatial coherence by associating each finer point with its closest superpoint.

Additionally, k-d trees are used in convolution layers within the model to define the receptive fields based on nearby superpoints. This enables efficient convolution operations over point clouds, preserving structural details while reducing the computational load [3].

By leveraging this grid-based subsampling with averaging, nearest-neighbor upsampling, and efficient k-d tree-based neighbor searching, we create a multiscale, hierarchical representation of the point cloud that retains important structural details while reducing data redundancy. This hierarchical structure enables efficient subsampling and upsampling operations, preserves spatial fidelity, and maintains computational efficiency.

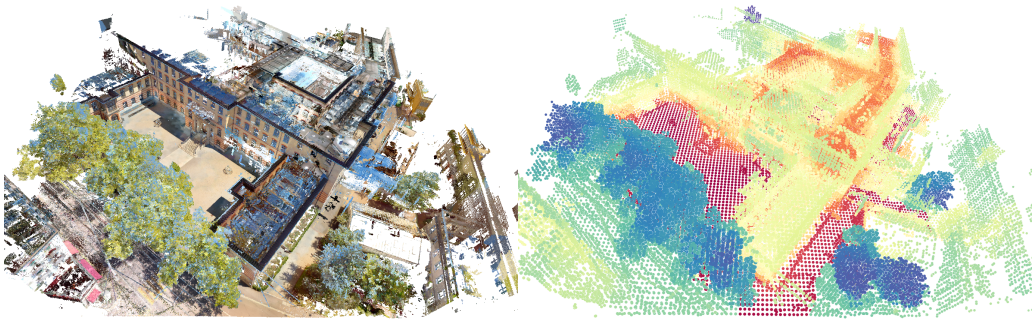


Figure 3: Visualization of a map and its extracted feature representation. The model learns to extract distinctive features from maps, which are visualized in a lower-dimensional space using t-SNE, illustrating the structural patterns and spatial relationships captured by the model.

4.2 Convolution for Point Clouds

To effectively learn from point clouds, we apply point cloud convolution techniques inspired by traditional image convolutions, specifically deformable kernel point convolutions [31, 48, 7]. Given a set of points $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^N$ in 3D space with associated features $F = \{\mathbf{f}_i\}_{i=1}^N$, where $\mathbf{p}_i \in \mathbb{R}^3$ and $\mathbf{f}_i \in \mathbb{R}^D$, the convolution of F by a kernel g at a point $\mathbf{x} \in \mathbb{R}^3$ is defined as:

$$(F * g)(\mathbf{x}) = \sum_{\mathbf{p}_i \in \mathcal{N}_x} g(\mathbf{p}_i - \mathbf{x})\mathbf{f}_i, \tag{12}$$

where $\mathcal{N}_x = \{\mathbf{p}_i \in \mathcal{P} \mid \|\mathbf{p}_i - \mathbf{x}\| \leq r\}$ is the radius neighborhood around $\mathbf{x}$ with a chosen radius r. This neighborhood ensures robustness to varying point densities by maintaining a consistent spherical domain for the function g, which helps the network learn meaningful representations.

In our hierarchical approach, the input and output points for each convolution layer are derived from the set of superpoints selected during subsampling. As each hierarchical level corresponds to a different voxel size, the neighborhood radius r for defining $\mathcal{N}_x$ is proportionally scaled according to the voxel size at that level. This proportional adjustment of r ensures that the receptive fields adapt to the spatial scale of the points, allowing the convolution to capture relevant structural information at each level.

4.2.1 Kernel Point Selection and Deformation

The critical component in this formulation is the definition of the kernel function g, where the uniqueness of kernel point convolution lies. The function g takes the neighbors' positions centered on $\mathbf{x}$ as input, denoted as $\mathbf{y}_i = \mathbf{p}_i - \mathbf{x}$. The domain of g is the ball $B_r^3 = \{\mathbf{y} \in \mathbb{R}^3 \mid \|\mathbf{y}\| \leq r\}$. Similar to image convolution kernels, g applies different weights to various areas within this domain using a set of kernel points.

Let $\{\tilde{\mathbf{x}}_k \mid k < K\} \subset B_r^3$ be the kernel points and $\{\mathbf{W}_k \mid k < K\} \subset \mathbb{R}^{D_{\text{in}} \times D_{\text{out}}}$ be the associated weight matrices that map features from dimension D_{in} to D_{out}. The kernel function g for any point $\mathbf{y}_i \in B_r^3$ is defined as:

$$g(\mathbf{y}_i) = \sum_{k<K} h(\mathbf{y}_i, \tilde{\mathbf{x}}_k)\mathbf{W}_k, \tag{13}$$

where $h(\mathbf{y}_i, \tilde{\mathbf{x}}_k)$ is the correlation between $\tilde{\mathbf{x}}_k$ and $\mathbf{y}_i$, which is higher when $\tilde{\mathbf{x}}_k$ is closer to $\mathbf{y}_i$. We use a linear correlation function inspired by bilinear interpolation:

$$h(\mathbf{y}_i, \tilde{\mathbf{x}}_k) = \max\left(0, 1 - \frac{\|\mathbf{y}_i - \tilde{\mathbf{x}}_k\|}{\sigma}\right), \tag{14}$$

where σ is the influence distance of the kernel points, chosen according to the input density.

This linear correlation, akin to rectified linear unit activations in neural networks, improves the efficiency of gradient backpropagation and simplifies the learning of kernel deformations.

To further enhance adaptability, each kernel point $\tilde{\mathbf{x}}_k$ undergoes a learned deformation, represented by an offset $\delta_k(\mathbf{x})$, which shifts the kernel point to better align with the local structure of the point cloud. This deformation allows the convolutional filter to flexibly capture fine geometric variations in the neighborhood. The deformed kernel location $\tilde{\mathbf{x}}'_k$ is given by:

$$\tilde{\mathbf{x}}'_k = \tilde{\mathbf{x}}_k + \delta_k(\mathbf{x}), \tag{15}$$

where $\delta_k(\mathbf{x})$ is the output of a learned kernel point convolution mapping D_{in} input features to $3K$ values. This deformation process enables the model to tailor each kernel point's position, adapting dynamically to the density, arrangement, and shape of points in each region of the cloud. By aligning kernel points with complex, uneven surfaces or varying densities, the convolution operation can more accurately capture local structural details, making it robust across diverse environments and point distributions.

4.2.2 Advantages

Our implementation of deformable kernel point convolutions offer several notable advantages when applied to point cloud data:

- **Geometric Adaptability**: By learning deformations of kernel point locations to align with local structures, the convolution operation becomes highly sensitive to the spatial distribution of points. This adaptability allows the model to capture fine-grained geometric details and effectively handle complex and irregular shapes within the point cloud.

- **Computational Efficiency via k-d Trees**: The implementation of k-d trees for neighbor searching significantly accelerates both kernel selection and convolution operations. This efficiency is crucial for processing large-scale point clouds, enabling rapid computation without compromising performance, and making the approach scalable to extensive datasets.

- **Preservation of Spatial Fidelity**: The use of a distance-based weighting function $h(y_i, \tilde{x}_k)$ ensures that points closer to the kernel centers have a greater influence on the convolution result. This emphasis on nearby points maintains spatial coherence in the feature extraction process, leading to more accurate and reliable representations of the point cloud's structure.

These advantages collectively empower deformable kernel point convolutions to extract robust and discriminative features from point clouds. By combining geometric flexibility, computational efficiency, and spatial consistency, this method enhances the model's ability to learn and represent complex three-dimensional environments effectively.

4.3 Feature Refinement with Geometric Transformers

Our feature refinement approach leverages a geometric attention mechanism that integrates spatial relationships among superpoints [50, 34]. By incorporating both positional and geometric context, the model achieves a robust, transformation-invariant representation that improves point cloud alignment.

- **Geometric Structure Embedding**: We apply a geometric structure embedding on superpoints to encode the transformation-invariant geometric structure of the superpoints. The core idea is to leverage the distances and angles computed with the superpoints, which are consistent across different point clouds of the same scene. Given two superpoints $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j \in \hat{\mathcal{P}}$, their geometric structure embedding consists of a pair-wise distance embedding and a triplet-wise angular embedding, described as follows:

  1. *Pair-wise Distance Embedding*: Given the distance between two superpoints,

  $$\rho_{i,j} = \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\|_2, \tag{16}$$

  the distance embedding $\mathbf{r}^D_{i,j}$ is computed by applying a sinusoidal function on $\rho_{i,j}/\sigma_d$, where σ_d is a hyper-parameter controlling the sensitivity to distance variations.

  2. *Triplet-wise Angular Embedding*: We compute the angular embedding using triplets of superpoints. First, we select the k nearest neighbors $\mathcal{K}_i$ of $\hat{\mathbf{p}}_i$. For each $\hat{\mathbf{p}}_x \in \mathcal{K}_i$, we compute the angle between vectors $\Delta_{x,i}$ and $\Delta_{j,i}$:

  $$\alpha^x_{i,j} = \arccos\left(\frac{\Delta_{x,i} \cdot \Delta_{j,i}}{\|\Delta_{x,i}\| \, \|\Delta_{j,i}\|}\right), \tag{17}$$

  where $\Delta_{x,i} = \hat{\mathbf{p}}_x - \hat{\mathbf{p}}_i$ and $\Delta_{j,i} = \hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i$. The triplet-wise angular embedding $\mathbf{r}^A_{i,j,x}$ is computed by applying a sinusoidal function on $\alpha^x_{i,j}/\sigma_a$, with σ_a controlling the sensitivity to angular variations.

  Finally, the geometric structure embedding $\mathbf{r}_{i,j}$ is computed by aggregating the pair-wise distance embedding and the triplet-wise angular embeddings:

  $$\mathbf{r}_{i,j} = \mathbf{r}^D_{i,j}\mathbf{W}^D + \max_x \left[\mathbf{r}^A_{i,j,x}\mathbf{W}^A\right], \tag{18}$$

  where $\mathbf{W}^D, \mathbf{W}^A \in \mathbb{R}^{d_t \times d_t}$ are projection matrices for the distance and angular embeddings, respectively. We use max pooling over x to improve robustness to variations in nearest neighbors due to self-occlusion.

- **Self-Attention Layers**: These layers model intra-cloud relationships, allowing each superpoint to attend to relevant superpoints within the same cloud. The attention score $e_{i,j}$ between superpoints $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_j$ is computed as:

  $$e_{i,j} = \frac{\left(f(\hat{\mathbf{p}}_i)\mathbf{W}^Q\right)\left(f(\hat{\mathbf{p}}_j)\mathbf{W}^K + \mathbf{r}_{i,j}\mathbf{W}^R\right)^T}{\sqrt{d_t}}, \tag{19}$$

  where $f(\hat{\mathbf{p}}_i)$ and $f(\hat{\mathbf{p}}_j)$ are input feature embeddings of superpoints $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_j$, $\mathbf{r}_{i,j}$ is the geometric structure embedding, and $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V, \mathbf{W}^R \in \mathbb{R}^{d_t \times d_t}$ are

the respective projection matrices for queries, keys, values, and geometric structure embeddings. The dimensionality d_t represents the size of the projected feature space, and is used here to normalize the attention score. The softmax operation is then applied to $e_{i,j}$ to obtain the attention weights $a_{i,j}$.

Finally, the output feature for each superpoint is computed as the weighted sum of all projected input features, where the attention weights $a_{i,j}$ determine the contribution of each feature.

- **Cross-Attention Layers**: The cross-attention layers facilitate interaction between the source and target point clouds by computing attention weights based on both feature and geometric similarities. For a superpoint $\hat{\mathbf{p}}_i$ in the source cloud and $\hat{\mathbf{q}}_j$ in the target cloud, the attention weights $\beta_{i,j}$ are computed similarly, incorporating cross-cloud geometric structure embeddings. This inter-cloud interaction enables the network to identify correspondences more accurately, even in cases of partial overlap, by focusing on regions with similar geometric structures.

This multi-stage refinement process focuses on progressively refining features by jointly leveraging local geometric structures and feature consistency across point clouds. These refined features subsequently enable the construction of a robust correspondence map, significantly enhancing the accuracy and efficiency of point cloud registration. As a result, the model can effectively handle complex 3D environments, improving both registration precision and robustness.

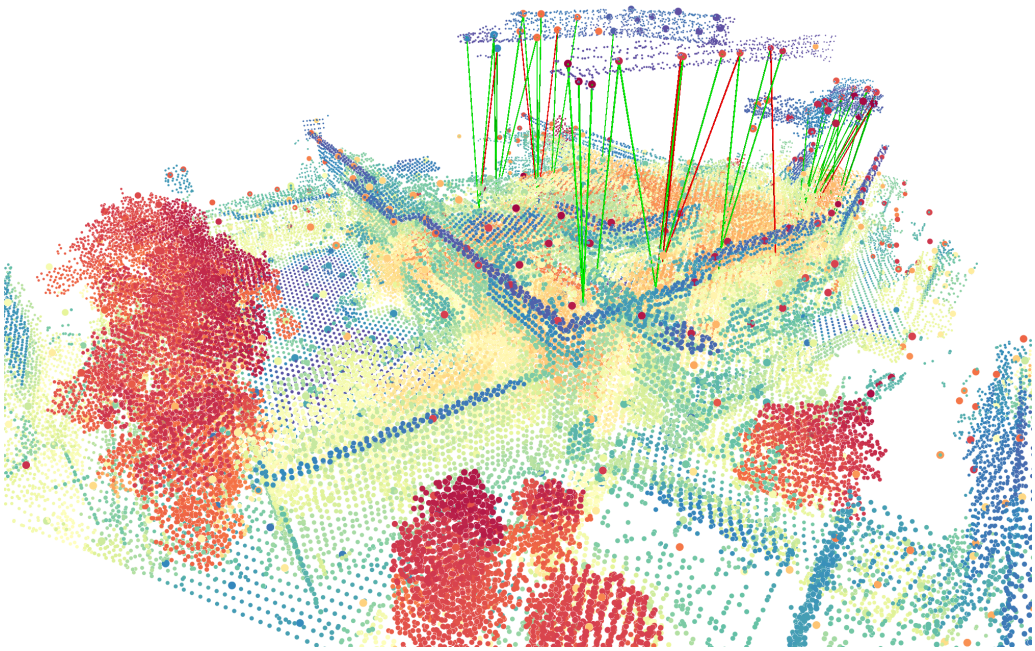


Figure 4: Visualization of superpoint matching results. Superpoints from the coarsest stage and fine points from the second stage are shown, colored using t-SNE. Green lines indicate correct matches with spatial overlap, red indicates wrong matches.

4.4 Correspondence Matching

Accurate correspondence matching is essential for point cloud registration. Our method enhances the matching process by incorporating both coarse and fine-grained matching stages to establish robust spatial coherence across features. This two-stage approach combines superpoint (coarse) matching with fine matching.

4.4.1 Superpoint Matching

In the first stage, we perform superpoint matching at a coarse level to establish initial correspondences between regions of the point clouds. We define one point cloud as the global map, with a set of superpoints $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_i\}$, and the other as the source or query point cloud, with its own set of superpoints $\hat{\mathcal{Q}} = \{\hat{\mathbf{q}}_j\}$. After iterating through the previous layers for N_t times, we obtain the hybrid feature matrices $\hat{\mathbf{H}}^{\mathcal{P}}$ for the global map and $\hat{\mathbf{H}}^{\mathcal{Q}}$ for the query point cloud.

We then normalize $\hat{\mathbf{H}}^{\mathcal{P}}$ and $\hat{\mathbf{H}}^{\mathcal{Q}}$ onto a unit hypersphere to ensure that the features are comparable in magnitude and direction. This normalization is achieved by applying ℓ_2-normalization to each feature vector:

$$\tilde{\mathbf{H}}_i^{\mathcal{P}} = \frac{\hat{\mathbf{H}}_i^{\mathcal{P}}}{\|\hat{\mathbf{H}}_i^{\mathcal{P}}\|_2}, \quad \tilde{\mathbf{H}}_j^{\mathcal{Q}} = \frac{\hat{\mathbf{H}}_j^{\mathcal{Q}}}{\|\hat{\mathbf{H}}_j^{\mathcal{Q}}\|_2}, \tag{20}$$

where $\tilde{\mathbf{H}}_i^{\mathcal{P}}$ and $\tilde{\mathbf{H}}_j^{\mathcal{Q}}$ are the normalized feature vectors for superpoints $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_j$, respectively.

Next, we compute the Gaussian correlation matrix $\mathbf{C}$ between the normalized features of the superpoints from the two point clouds:

$$C_{i,j} = \exp\left(-\frac{1}{2\sigma^2}\left\|\tilde{\mathbf{H}}_i^{\mathcal{P}} - \tilde{\mathbf{H}}_j^{\mathcal{Q}}\right\|_2^2\right), \tag{21}$$

where σ is a hyperparameter that controls the bandwidth of the Gaussian kernel, affecting sensitivity to feature differences.

To enhance the discriminative capability and mitigate the influence of feature magnitude variations, we apply a dual-normalization operation [38, 46] to the correlation matrix $\mathbf{C}$. This operation consists of two steps:

1. **Row-wise Normalization**: For each superpoint $\hat{\mathbf{p}}_i$ in the global map, we normalize the correlation scores across all superpoints in the query point cloud:

$$\bar{C}_{i,j} = \frac{C_{i,j}}{\sum_{j'} C_{i,j'} + \epsilon}, \tag{22}$$

   where ϵ is a small constant to prevent division by zero.

2. **Column-wise Normalization**: Subsequently, for each superpoint $\hat{\mathbf{q}}_j$ in the query point cloud, we normalize the row-normalized correlation scores across all superpoints in the global map:

$$\tilde{C}_{i,j} = \frac{\bar{C}_{i,j}}{\sum_{i'} \bar{C}_{i',j} + \epsilon}. \tag{23}$$

The resulting matrix $\tilde{\mathbf{C}}$ contains the dual-normalized correlation scores, emphasizing mutual correspondences while suppressing ambiguous matches. Only the top-k matches from the refined correlation matrix are retained at this stage, resulting in a set of superpoint correspondences $\hat{C}$ with the highest confidence scores, significantly reducing the computational load for the subsequent fine matching, laying a solid foundation for accurate correspondence estimation in subsequent steps.

4.4.2 Fine Matching

After establishing coarse matches between superpoints, we proceed to refine the correspondences at a finer scale by identifying dense point correspondences within each matched superpoint pair. For each superpoint correspondence $\hat{C}_i = (\hat{\mathbf{p}}_{x_i}, \hat{\mathbf{q}}_{y_i})$, we utilize an optimal transport layer inspired by [42] to extract local dense correspondences between the point sets $G^{\mathcal{P}}_{x_i}$ and $G^{\mathcal{Q}}_{y_i}$.

We define $G^{\mathcal{P}}_{x_i}$ as the *patch* of superpoint $\hat{\mathbf{p}}_{x_i}$, which consists of all points in the point cloud $\mathcal{P}$ that have $\hat{\mathbf{p}}_{x_i}$ as their closest superpoint. Formally,

$$G^{\mathcal{P}}_{x_i} = \left\{ \mathbf{p} \in \mathcal{P} \,\middle|\, \hat{\mathbf{p}}_{x_i} = \arg\min_{\hat{\mathbf{p}} \in \hat{\mathcal{P}}} \|\mathbf{p} - \hat{\mathbf{p}}\|_2 \right\}. \tag{24}$$

Similarly, $G^{\mathcal{Q}}_{y_i}$ is defined as the patch of superpoint $\hat{\mathbf{q}}_{y_i}$.

We first compute a cost matrix $\mathbf{C}_i \in \mathbb{R}^{n_i \times m_i}$ that measures the similarity between points in the matched superpoints:

$$\mathbf{C}_i = \frac{\mathbf{F}^{\mathcal{P}}_{x_i} (\mathbf{F}^{\mathcal{Q}}_{y_i})^\top}{\sqrt{\tilde{d}}}, \tag{25}$$

where $n_i = |G^{\mathcal{P}}_{x_i}|$ and $m_i = |G^{\mathcal{Q}}_{y_i}|$ are the numbers of points in the patches $G^{\mathcal{P}}_{x_i}$ and $G^{\mathcal{Q}}_{y_i}$, respectively. Here, $\mathbf{F}^{\mathcal{P}}_{x_i} \in \mathbb{R}^{n_i \times \tilde{d}}$ and $\mathbf{F}^{\mathcal{Q}}_{y_i} \in \mathbb{R}^{m_i \times \tilde{d}}$ are the feature matrices of the points within the patches, and $\tilde{d}$ is the feature dimensionality used for normalization.

To accommodate unmatched points and outliers, we augment the cost matrix $\mathbf{C}_i$ by adding an extra row and column filled with a learnable dustbin parameter α, resulting in an augmented cost matrix $\bar{\mathbf{C}}_i$:

$$\bar{\mathbf{C}}_i = \begin{bmatrix} \mathbf{C}_i & \mathbf{1}_{n_i}\alpha \\ \mathbf{1}^\top_{m_i}\alpha & \alpha \end{bmatrix}, \tag{26}$$

where $\mathbf{1}_{n_i}$ and $\mathbf{1}_{m_i}$ are vectors of ones with lengths n_i and m_i, respectively.

After this augmentation, we apply the Sinkhorn algorithm [6] to $\bar{\mathbf{C}}_i$ to compute a soft assignment matrix that approximates the optimal transport plan between the point sets. This process yields confidence scores for potential point correspondences, accounting for possible unmatched points via the dustbin parameter. We then extract the assignment matrix by removing the dustbin entries and select reliable point matches through mutual top-k selection. This strategy retains only the most confident and consistent correspondences, enhancing the robustness and accuracy of the fine matching process. By integrating these fine-level correspondences, the model can precisely align detailed structures within the point clouds, leading to improved registration performance.

4.5 Loss Functions

To optimize the model's performance, we apply tailored loss functions at different levels of matching. At the coarse level, we use an overlap-aware circle loss, which reweights matches based on the overlap ratio of points, giving higher weight to matches in areas with more overlapping points. This overlap-aware circle loss is defined as:

$$\mathcal{L}_{\text{circle}} = \sum_{i,j} \delta_{ij} \cdot \max(0, m - \|f(\mathbf{p}_i) - f(\mathbf{q}_j)\|)^2 \tag{27}$$

where δ_{ij} is an indicator function for overlapping regions, and m is a margin parameter that encourages correct matches while reducing the influence of distant or less relevant matches.

For finer levels, we use a negative log-likelihood loss to improve precision in the established correspondences. The fine-level loss function is defined as:

$$\mathcal{L}_{\text{fine}} = -\sum_{i,j} \log(S_{ij}) \tag{28}$$

where S_{ij} is the similarity score from the score matrix S between points $\mathbf{p}_i$ and $\mathbf{q}_j$. This loss function encourages highly accurate matches by penalizing misaligned correspondences, leading to robust alignment across varying conditions and levels of detail.

4.6 Fusion of RF Signals and Structural Features

To enhance the structural features extracted by the kernel point convolution layers, we integrate RF signal data using a fusion mechanism based on a graph neural network (GNN) with an attention mechanism [51]. This architecture effectively captures the propagation characteristics of RF signals, allowing for long-range information sharing across spatial points, enhancing feature distinctiveness, especially in complex environments where visual cues alone may be insufficient. Unlike traditional convolutional layers, this design enables information sharing over larger distances, which is particularly useful when only sparse RF measurements are available.

The fusion process starts with a self-attention mechanism applied independently to both the structural and RF features. The structural features are derived from the kernel point convolution, while the RF features are initialized as fingerprints $\mathbf{f}_i^{RF} = (\text{RSS}_1, \ldots, \text{RSS}_K)$, representing the received signal strengths (RSS) from multiple access points. Self-attention enables each feature set to propagate information across spatial points, enriching the representation even when RF measurements are sparse. By enhancing each feature's global contextual information, this process supports more robust registration and improves the system's ability to handle complex spatial relationships.

Next, we use a cross-attention layer to integrate the RF and structural features, allowing both modalities to reinforce each other's representation. This occurs in two steps:

- First, structural features serve as keys and values, while RF features act as queries, enabling the RF features to gather relevant structural context.

- Then, the roles are reversed: RF features become the keys and values, and structural features act as queries, reinforcing the structural features with RF-augmented context.

After multiple iterations of self- and cross-attention, a fused feature representation is generated. This fusion layer is strategically placed between feature extraction (Section 4.2) and feature refinement (Section 4.3), creating a highly informative feature map, suitable for precise localization and registration tasks in complex 3D environments.

On the ground truth map, where the locations of access points (APs) are known, we implement an RF embedding mechanism to enrich RF features with detailed propagation characteristics, enhancing the model's understanding of RF signal behavior. This embedding mechanism has three main components:

- **Distance Encoding**: Encodes the distance between an RF measurement point and APs within range, weighted by received signal strength (RSS), to reflect spatial separation.

- **Angular Encoding**: Uses sinusoidal encoding to capture the relative orientation between the RF measurement point and APs, accounting for directional signal propagation effects.

- **Fading Encoding**: Encodes the path loss exponent using a log-distance path loss model, capturing differential signal decay across space for improved sensitivity to positional variations.

These encodings collectively enhance the distinctiveness of the RF feature representations, enabling the model to learn highly informative representations.

4.7 Hierarchical Architecture and Feature Levels

Our model architecture is designed with a hierarchical framework, enabling efficient handling of point cloud data at different levels of granularity. By encoding and decoding features at multiple levels, we generate representations at varying resolutions, which allows for progressive refinement in registration tasks. This approach significantly reduces the number of points processed at each level, enhancing computational efficiency and supporting real-time performance.

Caching plays a critical role in this architecture by retaining features from previously processed layers, which minimizes recomputation during inference. These cached feature maps serve as dense representations of the environment and act as a form of compression, balancing accuracy and memory efficiency. Through this hierarchical and cached approach, we achieve computational efficiency while maintaining high precision in point cloud registration.

4.8 Training Data

We created a comprehensive point cloud registration benchmark by leveraging raw data from the LaMAR [43] dataset, which provides a rich and diverse set of environments and sensor data for robust training in point cloud registration.

4.8.1 Dataset Introduction

The LaMAR dataset is a valuable resource for training point cloud registration models due to its diversity and scale. This dataset includes recordings from three types of devices—laser scanners, AR headsets, and smartphones—spanning more than 100 hours of data collection and covering over 80,000 square meters across multiple real-world locations over a period of two years.

Why LaMAR is a Good Dataset:

- *Device Variety*: The dataset includes data from various sensor-equipped devices, such as laser scanners for precise depth measurements, smartphones, and AR headsets used as query devices. This diversity in sensor types and device usage patterns allows the model to generalize effectively across different hardware and operational contexts.

- *Environmental Diversity*: LaMAR captures a wide range of environments, both indoor and outdoor, featuring structural variations, changes in illumination, and long-term shifts in appearance.

- *Realistic Challenges*: The dataset includes realistic AR motion patterns and natural user movements, providing a challenging benchmark for localization and mapping models.

Weaknesses of Other Datasets: Other available datasets often suffer from several limitations that make them less suitable for comprehensive point cloud registration:

- *Ground Truth Inaccuracies*: Many datasets rely on ground truth data generated from Structure-from-Motion (SfM) or SLAM, which may introduce cumulative errors and reduce accuracy.

- *Lack of Large-Scale Scenes*: Most datasets are limited in scale and do not cover large, diverse environments, making it difficult for models to generalize.

- *Limited Sensor Diversity*: Predominantly, available datasets focus on driving scenes and lack variety in sensor types, leading to limited adaptability for AR and non-driving applications.

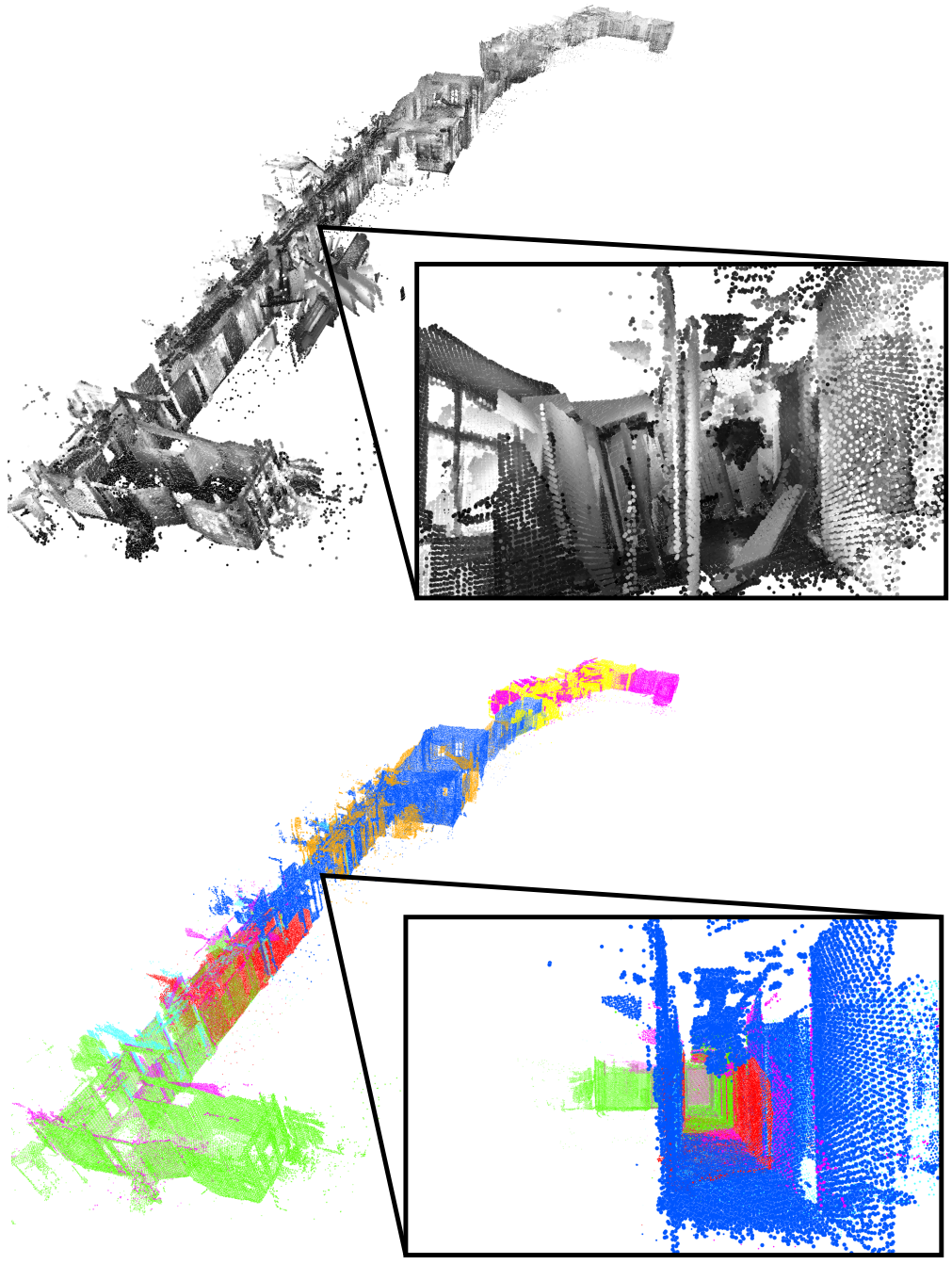


Figure 5: Visualization of improvements from our ground truth generation pipeline. Top: the original pipeline, where large jumps remain uncorrected. Bottom: our refined ground truth poses, with different colors representing distinct trajectory segments, separated at detected jumps.

4.8.2 Ground Truth Generation

To ensure precise ground truth for point cloud registration, we developed a robust ground truth generation pipeline using the raw data from the LaMAR dataset.

Ground Truth Map We used multiple laser scans to construct an accurate ground truth map. These laser scans were first aligned using image-based feature extraction, retrieval, and matching, followed by a lifting process to 3D using rendered depth maps. This initial alignment was refined through RANSAC and finally by Iterative Closest Point (ICP) to ensure high accuracy.

We denote ${}_A\mathbf{T}_B \in SE(3)$ as the 6-DoF pose, encompassing rotation and translation, required to transform a point in frame B to another frame A. For each pair of scanning sessions $(A, B) \in \mathcal{S}^2$, we first estimate a rigid transformation ${}_A\mathbf{T}_B$ to align session B with session A. To do this, we select each image I_i^A in A and identify its r most similar images $\{I_j^B\}_{1\leq j\leq r}$ in B based on global image descriptors [22, 1, 35]. This helps determine a large-scale registration between sessions.

Next, we extract sparse local image features and establish 2D-2D correspondences $\{\mathbf{p}_i^A, \mathbf{p}_j^B\}$ for each image pair (i, j). The 2D keypoints $\mathbf{p}_i \in \mathbb{R}^2$ are then lifted to 3D, $\mathbf{P}_i \in \mathbb{R}^3$, by tracing rays through the dense mesh of the corresponding session. This process yields 3D-3D correspondences $\{\mathbf{P}_i^A, \mathbf{P}_j^B\}$, from which we estimate an initial relative pose using RANSAC [12] to reject outliers.

The initial pose is further refined using the point-to-plane Iterative Closest Point (ICP) algorithm [4] applied to the pairs of lidar point clouds. This refinement step enhances accuracy by aligning points based on both position and surface orientation.

By using state-of-the-art local image features that can handle drastic illumination and viewpoint changes [41, 9, 37], and combining them with the strong geometric constraints of the registration, the system achieves robust alignment over long-term temporal changes without manual initialization. This approach has successfully registered building-scale scans captured over intervals exceeding a year, even with significant structural changes in the environment.

Query Session GT Pose Alignment For the query sessions (data collected from smartphones and AR headsets), we initially tested the ground truth pose generation pipeline proposed in the LaMAR benchmark. Their pipeline relied on image-based localization for an initial alignment of relative poses, followed by a voting mechanism to merge device poses, and further refinement using pose graph optimization and bundle adjustment. However, our testing revealed significant limitations, as the initial relative poses from the devices were frequently inaccurate, particularly in cases of large rotational or translational shifts.

Our Improved Ground Truth Generation Pipeline: To accurately register each AR sequence into the dense ground truth reference model $\mathcal{M}$, we designed a new ground truth generation pipeline based on a per-frame absolute pose estimation. Given a sequence of n frames, we introduce a method that estimates the absolute pose $\{{}_w\mathbf{T}_i\}_{1\leq i\leq n}$ for each frame i, where i is a common reference world frame, and each frame corresponds to an image captured at a specific time.

Inputs: We assume access to visual-inertial tracker trajectories $\{ {}_0\mathbf{T}_i^{\text{track}}\}$ generated by ARKit for iPhone/iPad and the on-device tracker for HoloLens. For simplicity, we use $\mathbf{T}_i$ instead of ${}_0\mathbf{T}_i^{\text{track}}$ where the meaning is clear and no ambiguity arises.

To address existing limitations in the initial relative poses, we designed the following steps in our pipeline:

1. **Trajectory Jump Detection**: We detected abrupt changes, or "jumps", in the device trajectories by analyzing the rotational and translational speeds between consecutive poses. Given consecutive poses $\mathbf{T}_i$ and $\mathbf{T}_{i+1}$, the rotational change $\Delta\theta_i$ and translational change $\Delta\mathbf{t}_i$ are computed as:

$$\Delta\theta_i = \arccos\left(\frac{\text{trace}(\mathbf{R}_i^\top \mathbf{R}_{i+1}) - 1}{2}\right), \quad \Delta\mathbf{t}_i = \|\mathbf{t}_{i+1} - \mathbf{t}_i\|_2, \tag{29}$$

   where $\mathbf{R}_i$ and $\mathbf{t}_i$ are the rotation matrix and translation vector of pose $\mathbf{T}_i$. We identified a jump if $\Delta\theta_i$ or $\Delta\mathbf{t}_i$ exceeded predefined thresholds θ_{th} and t_{th}, respectively.

2. **Session Segmentation**: We segmented each query session into smaller, continuous subsections $\{S_k\}$ at the detected jumps, isolating segments with smoother motion conducive to accurate alignment.

3. **Image-Based Localization**: For each segmented query S_k, we perform an initial localization by retrieving a fixed number r of relevant reference images $\{I_j^{\text{ref}}\}_{1\le j\le r}$ from the ground truth map $\mathcal{M}$, using global image descriptors to select the most similar images for each query frame I_i^{query}. We then match sparse local features extracted from each query frame to each retrieved reference image, establishing 2D-2D correspondences $\{\mathbf{p}_i^{\text{query}}, \mathbf{p}_j^{\text{ref}}\}$. These 2D keypoints in the reference images are lifted to 3D to form a set of 2D-3D correspondences $\{\mathbf{p}_i^{\text{query}}, \mathbf{P}_j^{\text{ref}}\}$, which are used for pose estimation.

4. **Gateway Voting**: The goal is to find the transformation ${}_0\mathbf{T}_W$ that best aligns each frame with the ground truth map by evaluating inlier consistency among the set of candidate poses $\{ {}_i\mathbf{T}_W\}_{0\le i\le r-1}$, calculated based on the 2D-3D correspondences. This consistency is quantified by counting inliers.

   Inlier Count Calculation for Candidate Poses:

   For each timestamp i in the set of candidate poses $\{ {}_i\mathbf{T}_W\}_{0\le i\le r-1}$:

   1. For all other timestamps $j \in \{0, \ldots, r-1\} \setminus \{i\}$, compute a candidate pose-based hypothesis:

$$ {}_j\mathbf{T}_W^{\text{hyp}} = {}_i\mathbf{T}_W \, {}_0\mathbf{T}_i \, {}_0\mathbf{T}_j^{-1}$$

   2. Compute the alignment loss as the Frobenius norm of the difference between the hypothesized transformation ${}_j\mathbf{T}_W^{\text{hyp}}$ and the candidate pose ${}_j\mathbf{T}_W$:

$$\text{loss} = \left\| {}_j\mathbf{T}_W^{\text{hyp}} - {}_j\mathbf{T}_W \right\|_F$$

   3. If this alignment loss is below a predefined threshold τ, consider j an inlier for

timestamp i, and increment the inlier count for i.

The inlier count for each timestamp i is given by:

$$\text{count}(i) = \sum_{j \neq i} \begin{cases} 1, & \text{if } \left\| {}_j\mathbf{T}_W^{\text{hyp}} - {}_j\mathbf{T}_W \right\|_F < \tau \\ 0, & \text{otherwise} \end{cases}$$

Final Transformation:

The timestamp i^* with the highest inlier count $\text{count}(i^*)$ is selected as the one that provides the most consistent alignment with the ground truth map. The corresponding transformation ${}_0\mathbf{T}_W$ for this timestamp i^* is used as the optimal transformation from the device's world coordinate system to the ground truth:

$${}_0\mathbf{T}_W = {}_{i^*}\mathbf{T}_W \, {}_0\mathbf{T}_{i^*}$$

The inlier ratio, indicating the consistency level across frames, is calculated as:

$$\text{inlier ratio} = \frac{\text{count}(i^*)}{r}$$

where r is the total number of candidate poses. This inlier-based voting mechanism enhances robustness by ensuring that the chosen transformation has maximum alignment across multiple localized frames.

5. **Point Cloud Generation**: Using depth information, we reconstruct point clouds $\mathcal{Q}_k$ for each segment S_k. For each pixel (u, v) in the depth image D_i corresponding to pose $\mathbf{T}_i$, we back-project the depth $d_i(u, v)$ to obtain the 3D point $\mathbf{p}_i(u, v)$:

$$\mathbf{p}_i(u, v) = \mathbf{T}_i \left(d_i(u, v)\mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right), \tag{30}$$

where $\mathbf{K}$ is the camera intrinsic matrix.

6. **Multi-Scale ICP Refinement**: First, we apply the initial transformation ${}_0\mathbf{T}_W$ obtained from gateway voting to the point cloud $\mathcal{Q}_k$, aligning it to the ground truth coordinate system W. Then, we use point-to-plane ICP refinement at multiple scales to minimize any residual alignment errors.

   After refinement, we compose the final ground truth transformation ${}_0\mathbf{T}_W^{\text{final}}$ by sequentially applying the initial transformation followed by the ICP adjustment:

$${}_0\mathbf{T}_W^{\text{final}} = {}_0\mathbf{T}_W \, \mathbf{T}^{\text{refine}} \tag{31}$$

After running this refined pipeline, we filtered out low-quality sessions, retaining only those with a high number of initially localized images and a reliable gateway pose inlier ratio. This process significantly improved the accuracy and robustness of the ground truth data, providing high-quality training data for point cloud registration tasks.

Model	Time (ms)	RTE (m)	RRE (°)	Recall (%)
HLoc	**19**	0.58	1.67	72
Ours (10m)	43	0.97	3.14	19
Ours (20m)	48	0.78	2.01	38
Ours (30m)	51	0.59	2.42	67
Ours (40m)	53	**0.45**	**1.47**	**86**

Table 1: Evaluation results on our point cloud registration benchmark.

5 Results

5.1 Baseline Method

To evaluate the performance of our model, we compared it against the state-of-the-art HLoc [41] using a specific configuration optimized for large scenes. While more robust implementations, using chunk alignment, exist, they lead to incomparable inference times. Therefore, we selected this configuration to ensure a fair and feasible comparison. This setup employs the SuperPoint feature extractor and SuperGlue matcher with parameters as follows:

The SuperPoint [9] features are extracted with non-maximum suppression (NMS) radius set to 3 and a maximum of 2048 keypoints per image. Images are preprocessed in grayscale with a maximum resize dimension of 1024 pixels. SuperGlue [42] is used for matching with weights optimized for large scale environments and employs 5 Sinkhorn iterations to refine the matching.

For retrieval, a fusion method combines NetVLAD [1] and AP-GeM [36] descriptors. NetVLAD preprocesses images with a maximum resize dimension of 640 pixels, while AP-GeM utilizes a similar resize setting. In the mapping step, an additional filtering mechanism enforces constraints on frustum depth (up to 20 meters), rotation (up to 120 degrees), and translation (up to 20 meters) to improve match quality. The final mapping pairs are filtered with parameters for frustum and pose, with a maximum of 10 pairs selected.

Using this configuration as a benchmark, we assess our model's accuracy and efficiency in establishing correspondences and achieving robust localization.

5.2 Configuration

We utilized a five-stage extraction process, where the voxel size on the coarsest level was set to 8 meters. To ensure robustness, we initialized points without any visual features, prioritizing generalization across different environments rather than relying on visual cues. Instead, we initialized the first level with a convolution on the input points, allowing the model to capture initial structural information directly from the spatial distribution of input points. The superpoint matching was performed at the coarsest level, while the fine matching was carried out on the second level with a voxel size of 1 meter.

Distance	CIR (%)
10 m	34
20 m	53
30 m	69
40 m	89

Table 2: CIR (Coarse Inlier Rate) represents the percentage of superpoints successfully matched, indicating the fraction of matches where the matched superpoints have actual overlap at various distances.

5.3 Training

The initial training phase was conducted on low-scale maps, enabling efficient training without the need to cache data, such as neighbor lists of the ground truth map or downsampling lists. This setup allowed for greater data augmentation, leading to a better initial feature representation. As training progressed, we scaled up the map sizes, introducing more negative samples for the superpoint matching, enhancing the model's ability to distinguish between similar locations. However, due to GPU memory limitations, the model could not be trained on large maps. To address this, we cropped out and randomly shifted the surrounding map segment to ensure the inclusion of positive samples and randomly selected other parts of the map to provide a diverse array of negative samples.

5.4 Inference

During inference, we reduced the search space by leveraging Wi-Fi signal data. Specifically, we calculated the cumulative sum of the reciprocal of the absolute RSSI (dBm) for each MAC address in the query recording to estimate signal strength. Starting with the strongest signals, we progressively identified regions in the ground truth map where each MAC address was detected. This process continued until the cumulative area covered reached at least 60 meters by 60 meters, ensuring high overlap. Finally, we conducted inference over various time windows based on the past 10, 20, 30, and 40 meters of point cloud data to evaluate the model's adaptability across different spatial scales.

5.5 Performance Comparison

Table 1 presents a comparison of our model against HLoc on various metrics, including inference time, Relative Translation Error (RTE), Relative Rotation Error (RRE), and recall. Recall is defined as the percentage of cases where RTE is below 2 meters and RRE is below 5 degrees. RTE and RRE are computed exclusively for poses that were successfully recalled. Note that both models utilized cached map features, and the time measurements exclude the final pose estimation process, as it is similar for both methods. The results indicate that increasing the search distance in our model improves accuracy, with lower RTE and RRE values, and boosts recall. Using the previous 40 meters of recorded point cloud data, our model achieves the highest performance in terms of precision and recall, albeit with a higher computational cost.

6 Conclusion and Discussion

In this work, we introduced a robust approach to localization by treating it as a point cloud registration problem, effectively addressing limitations in traditional, image-based localization methods. By leveraging the structural and geometric properties of 3D point clouds and integrating multimodal data through RF signals, our model achieves enhanced localization precision even in environments that are challenging for visual sensors, such as low-light or highly repetitive structural settings. Through the hierarchical feature-matching architecture and use of transformation-invariant geometric features, the model demonstrates adaptability across complex indoor and outdoor environments.

Our method shows competitive results compared to the state-of-the-art HLoc framework, with recall rates and accuracy metrics that validate its capability to handle large-scale and dynamic environments. Notably, by using a layered approach to point cloud feature extraction, caching strategies, and trajectory-based matching, our model maintains real-time performance while minimizing computational overhead. These design choices make it not only a strong option for traditional localization applications but also a promising candidate for use in real-time and resource-constrained scenarios, such as on mobile devices or embedded systems.

Looking ahead, our approach opens up several promising avenues for future work:

1. **Self-supervised Representation Learning**: Given the effectiveness of our model in learning distinctive spatial features, we propose using self-supervised learning to refine these features without requiring extensive labeled data. This approach could enable the model to adapt to new environments autonomously, supporting broader deployment across diverse, unstructured settings.

2. **Adaptive Hierarchical Matching for Scalability**: While our hierarchical framework demonstrates robustness, further optimizing the depth and adaptability of the matching process could yield computational benefits, particularly in large-scale maps. Future work could investigate dynamic scaling within the hierarchy based on environmental complexity, thereby tailoring the model's resource usage to the environment's specific demands.

3. **Real-World Deployments and Continuous Mapping**: Deploying the model in real-world settings, such as for service robots, and enabling it to continuously update and refine the point cloud map in real time would provide valuable testing opportunities. Real-time feedback loops would allow the robot to adapt to changes in the environment, like moving furniture or new obstacles, demonstrating the model's robustness and adaptability in dynamic settings.

4. **Representation Learning for Downstream Tasks**: The distinctive representations generated by our model highlight its potential as a foundational tool for representation learning. By optimizing localization with a matching loss, the model learns highly discriminative features that can be valuable for various downstream tasks beyond point cloud alignment. These learned embeddings could serve as pre-trained models for tasks requiring a strong geometric and spatial understanding, such as object detection, scene reconstruction, or semantic segmentation, thereby extending the model's utility and adaptability across diverse applications.

In summary, our proposed model advances the field of localization by offering a robust and adaptable alternative to purely visual approaches. By building upon these results and exploring the future directions outlined above, we believe our work contributes a foundational framework for developing high-precision, real-time localization solutions across increasingly complex and diverse environments. This work lays the groundwork for further innovations in multimodal data integration, autonomous navigation, and adaptive localization in both controlled and dynamic real-world applications.

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[2] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987.

[3] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18:509–517, 1975.

[4] Yang Chen and Gérard G. Medioni. Object modeling by registration of multiple range images. *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 2724–2729 vol.3, 1991.

[5] Mark Cummins and Paul Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *I. J. Robotic Res.*, 27:647–665, 06 2008.

[6] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transportation distances, 2013.

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks, 2017.

[8] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 2, pages 1322–1328 vol.2, 1999.

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[10] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint detection and description of local features, 2019.

[11] Christine Evers and Patrick A. Naylor. Optimized self-localization for slam in dynamic scenes using probability hypothesis density filters. *IEEE Transactions on Signal Processing*, 66(4):863–878, 2018.

[12] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981.

[13] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014.

[14] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte carlo localization: Efficient position estimation for mobile robots. pages 343–349, 01 1999.

[15] Dorian Gálvez-López and Juan D. Tardós. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28:1188–1197, 2012.

[16] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval, 2017.

[17] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140:107–113, 1993.

[18] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2432–2437, 2005.

[19] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006.

[20] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap, 2021.

[21] Muhammad Ahmed Humais, Mohammad Chehadeh, Rana Azzam, Igor Boiko, and Yahya Zweiri. Vistune: Auto-tuner for uavs using vision-based localization. *IEEE Robotics and Automation Letters*, 9:9111–9118, 2024.

[22] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, 2010.

[23] Lik-Hang Lee, Tristan Braud, Pengyuan Zhou, Lin Wang, Dianlei Xu, Zijun Lin, Abhishek Kumar, Carlos Bermejo, and Pan Hui. All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda. *ArXiv*, abs/2110.05352, 2021.

[24] Keith Y. K. Leung, Felipe Inostroza, and Martin Adams. An improved weighting strategy for rao-blackwellized probability hypothesis density simultaneous localization and mapping. In *2013 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pages 103–110, 2013.

[25] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[26] R.P.S. Mahler. Multitarget bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, 2003.

[27] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. Fastslam: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth National Conference on Artificial Intelligence*, page 593–598, USA, 2002.

American Association for Artificial Intelligence.

[28] Thien-Minh Nguyen, Shenghai Yuan, Muqing Cao, Thien Hoang Nguyen, and Lihua Xie. Viral slam: Tightly coupled camera-imu-uwb-lidar slam, 2021.

[29] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I, 2004.

[30] Stefanos Pertigkiozoglou, Evangelos Chatzipantazis, and Kostas Daniilidis. Biequiformer: Bi-equivariant representations for global point cloud registration, 2024.

[31] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017.

[32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[33] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, August 2018.

[34] Zheng Qin, Hao Yu, Changjian Wang, Yulan Guo, Yuxing Peng, and Kai Xu. Geometric transformer for fast and robust point cloud registration, 2022.

[35] Jerome Revaud, Jon Almazan, Rafael Sampaio de Rezende, and Cesar Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss, 2019.

[36] Jerome Revaud, Jon Almazan, Rafael S. Rezende, and Cesar Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[37] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2d2: Repeatable and reliable detector and descriptor, 2019.

[38] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood consensus networks, 2018.

[39] Antoni Rosinol, John J. Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444, 2022.

[40] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011.

[41] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From

coarse to fine: Robust hierarchical localization at large scale, 2019.

[42] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks, 2020.

[43] Paul-Edouard Sarlin, Mihai Dusmanu, Johannes L. Schönberger, Pablo Speciale, Lukas Gruber, Viktor Larsson, Ondrej Miksik, and Marc Pollefeys. LaMAR: Benchmarking Localization and Mapping for Augmented Reality. In *ECCV*, 2022.

[44] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015.

[45] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477, 2003.

[46] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers, 2021.

[47] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization, 2020.

[48] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds, 2019.

[49] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents series. MIT Press, 2005.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[51] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[52] Ba-Ngu Vo, Samiksha Singh, and Arnaud Doucet. Sequential monte carlo implementation of the phd filter for multi-target tracking. volume 2, pages 792– 799, 02 2003.

[53] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12776–12786, 2021.