Autonomous Robots (2026) 50:14
https://doi.org/10.1007/s10514-025-10232-5

Visual localization in 3D maps: comparing point cloud, mesh, and NeRF representations

Lintong Zhang[1] · Yifu Tao[1] · Jiarong Lin[2] · Fu Zhang[2] · Maurice Fallon[1]

Received: 19 October 2024 / Revised: 16 September 2025 / Accepted: 24 November 2025 / Published online: 10 March 2026

Abstract

Recent advances in mapping techniques have enabled the creation of highly accurate dense 3D maps during robotic missions, such as point clouds, meshes, or NeRF-based representations. These developments present new opportunities for reusing these maps for localization. However, there remains a lack of a unified approach that can operate seamlessly across different map representations. This paper presents and evaluates a global visual localization system capable of localizing a single camera image across various 3D map representations built using both visual and lidar sensing. Our system generates a database by synthesizing novel views of the scene, creating RGB and depth image pairs. Leveraging the precise 3D geometric map, our method automatically defines rendering poses, reducing the number of database images while preserving retrieval performance. To bridge the domain gap between real query camera images and synthetic database images, our approach utilizes learning-based descriptors and feature detectors. We evaluate the system's performance through extensive real-world experiments conducted in both indoor and outdoor settings, assessing the effectiveness of each map representation and demonstrating its advantages over traditional structure-from-motion (SfM) localization approaches. The results show that all three map representations can achieve consistent localization success rates of 55% and higher across various environments. NeRF synthesized images show superior performance, localizing query images at an average success rate of 72%. Furthermore, we demonstrate an advantage over SfM-based approaches that our synthesized database enables localization in the reverse travel direction which is unseen during the mapping process. Our system, operating in real-time on a mobile laptop equipped with a GPU, achieves a processing rate of 1 Hz.

Keywords Localization · Mapping · Sensor fusion · RGB-D perception

1 Introduction

Recognizing a previously visited place and estimating a robot/sensor's accurate metric pose is fundamental to the problem of robot localization. Localization not only addresses the question of determining where a robot is, but it also enables other tasks such as loop detection in Simultaneous Localization and Mapping (SLAM), multi-robot or multi-session mapping, and augmented reality. Robot localization can be achieved using various sensor modalities, with camera-based (Sarlin et al., 2019; Sattler et al., 2017; Zeisl et al., 2015) and lidar-based approaches (Kim & Kim, 2018; Vidanapathirana et al., 2022; Uy & Lee, 2018) being the most extensively studied. While cameras offer a compact form factor and are often low-cost, visual localization and mapping systems often struggle to build large scale accurate 3D maps, especially in real time. Conversely, lidar systems are physically bigger, have higher power

✉ Lintong Zhang
lintong@robots.ox.ac.uk

Yifu Tao
yifu@robots.ox.ac.uk

Jiarong Lin
zivlin@buaa.edu.cn

Fu Zhang
fuzhang@hku.hk

Maurice Fallon
mfallon@robots.ox.ac.uk

[1] Department of Engineering Science, University of Oxford, Pok Fu Lam, Hong Kong, China

[2] Department of Mechanical Engineering, University of Hong Kong, Wellington Square, Oxford, UK

consumption and increased cost—factors which pose challenges for their widespread deployment on mobile robots. Because of this, researchers have proposed combining the advantages of both sensors by constructing a 3D dense map using lidar data (potentially aided by cameras), and then performing localization using only the camera (Wolcott & Eustice, 2014; Pascoe et al., 2015).

For any localization system, a prior representation of the scene—a prior map—is required. The prior map is usually built using the same sensor modality that will be used for later deployment. For example, autonomous industrial inspection has been demonstrated using legged platforms equipped with lidar (Gehring et al., 2021), where a prior map is built by accumulating laser scans, and localization is performed against the map using of ICP (Besl & McKay, 1992). However, lidar localization is sensitive to physical scene changes and relies on a good initial estimate for ICP registration. Conversely, purely visual localization methods typically build a map using a structure-from-motion pipeline (SfM) such as COLMAP (Schönberger & Frahm, 2016), and then localize within it using visual place recognition followed pose refinement using methods such as perspective-n-points (PnP) (Sarlin et al., 2019). SfM maps could, however, be inflexible as they are constrained to the local descriptors used to build them. The pose estimated achieved by visual localization within a SfM map are not scaled to real-world dimensions. As a result, additional information is required for effective deployment in robotic applications. Furthermore, these methods are generally not real-time as the SfM bundle adjustment process can take anywhere from a few hours to an entire day, depending on the size of the images or the map.

Cross-modal localization is typically more challenging due to the fundamental differences between sensing modalities and map representations. LiDAR-based approaches often rely on point cloud maps, whereas some applications may only provide a colored mesh map (Yin et al., 2023). In contrast, visual localization methods rely on a database of 3D points with associated descriptors, specifically tailored to the feature detectors used during the map construction (Zeisl et al., 2015; Sattler et al., 2017). Moreover, emerging scene representations such as NeRF (Liu et al., 2023) and Gaussian Splatting (Matsuki et al., 2024) are also dependent on the specifics of the original sensing setup. The variety of map representations poses challenges when applying existing localization approaches.

There has been many recent advances in high-quality 3D color mapping—using either mobile SLAM or fixed Terrestrial Laser Scanners (TLS) data. This has creates new avenues for research into techniques which repurpose these maps for robotic localization. We propose to create a unified synthetic map representation/database that accommodates the different types of maps produced by these new mapping systems. In particular, these 3D mapping systems enable the rendering of novel synthetic images from arbitrary rendering positions. In our approach, the visual localization database is made up of rendered RGB and depth images at chosen poses and can be automatically synthesized from color point clouds, meshes, or NeRF maps. These 3D color maps can be built using vision and lidar data acquired from either industrial-grade TLS or hand-held mapping devices. Using this unified representation, we achieve scalable global localization in varied environments, resilient to scene changes and lighting variations with visual learning based components. An overview of the system operation is shown in Fig. 1. In addition, we perform a direct comparison with SfM methods to demonstrate the capability of synthesizing views in the opposite direction of travel, as well as highlighting the advantage of reducing the number of required database images. All experiments are conducted in real-world environments, and the corresponding datasets and ground truth maps are available as part of the Oxford Spires Dataset (Tao et al., 2025) .[1].

The key contributions of our work include:

- A versatile and unified approach for global visual localization of a single camera in dense 3D maps, using a database of synthetic RGB and depth images generated from point cloud, mesh, or NeRF representations.
- A strategy for automatically determining the poses within the color 3D maps from which the synthetic images should be generated.
- An extensive evaluation across both indoor and outdoor environments comparing visual localization performance across different map representations.
- A performance comparison between our localization system with synthetic images and two (purely) visual localization systems with real images.

The proposed system runs on a mobile laptop at 1 Hz without extensive optimization and is suitable for use as a place recognition module within SLAM systems.

2 Related work

We define localization as the process of estimating the 6 DoF pose of a sensor within a 3D prior map or database. This is commonly achieved by means of a place recognition stage—which finds place candidates in the prior map—, and a registration step—that estimates a precise 6 DoF pose using the candidates. Given two sensing inputs of vision and

[1] https://dynamic.robots.ox.ac.uk/datasets/oxford-spires/

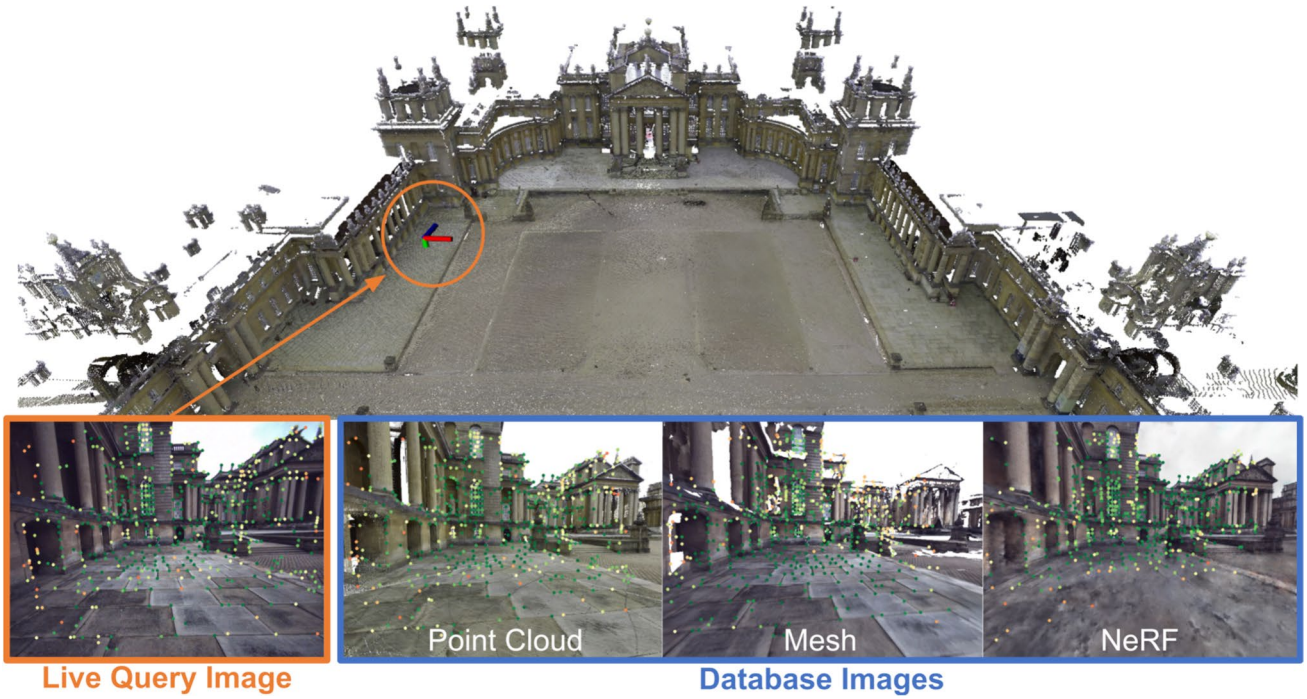


Fig. 1 Localization of a single query image to a database of images that are synthesized from either point cloud, mesh, or NeRF representations of Blenheim Palace. After identifying a matching image in the database, features are extracted with SuperPoint (as shown above) and matched with SuperGlue. (Displayed point cloud is not directly used.)

lidar, the current literature can be categorized into four categories—based on sensing modality. These include (a) visual localization within a visual map, (b) lidar localization within a lidar prior map, (c) combined visual and lidar localization within a joint visual and lidar map, and (d) cross-modality visual localization within a lidar prior map or vice versa.

2.1 Visual localization

Visual localization methods aim to obtain a 6 DoF pose by localizing against a map built from multiple images. In general, such maps are built using Structure-from-Motion (SfM) pipelines (Schönberger & Frahm, 2016), which are able to recover the 3D structure from multiple views. Given that this reconstruction is accurate up to a scale factor, additional prior information, such as knowledge of a stereo baseline or direct depth sensing, is required for metric localization.

For localization, visual place recognition is implemented as an image retrieval problem. Classic methods such as FAB-MAP (Cummins & Newman, 2008) and DBoW (Galvez-López & Tardos, 2012) solve the problem using local features, but more recent methods such as NetVLAD (Arandjelović et al., 2018), PatchNetVLAD (Hausler et al., 2021), EigenPlaces (Berton et al., 2023), MixVPR (Ali-bey et al., 2023) rely entirely on neural architectures to obtain global descriptors for retrieval. Metric pose estimation is then performed via local feature matching using approaches such as SIFT (Lowe, 2004), SuperPoint (DeTone et al., 2018), or R2D2 (Revaud et al., 2019). This type of pipeline has been implemented in methods such as HLoc (Sarlin et al., 2019) and Schönberger and Frahm (2016), which employ a hierarchical approach to large-scale visual localization.

In addition to the aforementioned methods, image rendering based approaches have been explored for visual localization in more recent studies. MeshLoc (Panek et al., 2022) demonstrates it is feasible to construct a dense 3D mesh model from multi-view stereo point clouds and to render synthetic images via an OpenGL pipeline to create a visual database. Works by Trivigno et al. (2024) and Zhang et al. (2021) employed an iterative strategy to render images and to refine the pose set based on a query image. While effective for single-query localization, these methods rely on an initial pose estimate and lack the real-time performance required for robotic applications.

With the emergence of NeRF rendering technique, NeRF-Loc (Liu et al., 2023) introduced an end-to-end localization framework that employs a conditional NeRF as the 3D representation, using an RGB–depth image pair as input. In contrast, Loc-NeRF (Maggio et al., 2023) first trains a NeRF model of the scene and subsequently applies a more traditional Monte Carlo–based approach for localization.

2.2 Lidar localization

Similar to visual localization, lidar localization also maintains a map database—in this case of 3D lidar scans. A hierarchical approach to retrieval and matching typically relies on generating global descriptors from each lidar scan. Early work started with handcrafted descriptors. Himstedt et al. (2014) draw inspiration from the bag-of-words approach and transform 2D lidar scans into a histogram representation for place recognition. He et al. (2016) project the 3D point cloud into multiple 2D planes, generating descriptors for each of the planes based on point density, and combining them into a global descriptor. More recent work has moved towards learned global descriptors. Vidanapathirana et al. (2022) and Uy and Lee (2018) both leverage deep learning networks and are able to produce robust global descriptors for loop closure in outdoor large-scale environments.

Recent work in lidar localization techniques has increasingly integrated semantics and extracted segments within the lidar point cloud. Kong et al. (2020) introduce a place recognition approach based on a semantic graph, preserving the topological information of the point cloud. They show that by working on a semantic level, their method can be more robust to environmental changes. Aiming to address the same challenge, Segmap (Dubé et al., 2020) introduces a segment-based map representation and generates descriptors for these segments using a learned network. These descriptors serve multiple purposes: localization, map reconstruction, and semantic information extraction. Building upon the segment concept, Locus (Vidanapathirana et al., 2020) further extends the approach by combining descriptors of all segments with spatio-temporal high-order pooling to generate fixed-size global descriptors, which are effective for place recognition. This method was demonstrated to achieve robustness in challenging scenarios such as viewpoint changes and occlusions. However, there are unique challenges for indoor environments, as objects can clutter the room in confined spaces, leading to more occlusions. InstaLoc (Zhang et al., 2023) demonstrated good performance for indoor localization by extracting object instances and generating a descriptor for each object before globally matching these objects to estimate the pose.

More closely related to our work, Chen et al. (2021) proposed an approach that exploits range images generated from 3D lidar scans. Their method first constructs a triangular mesh of the environment and then generates synthetic range images from the mesh map to form a database. This database is then used as part of a Monte Carlo localization system, demonstrating strong performance in autonomous driving scenarios.

2.3 Combined visual/lidar localization

In scenarios where multiple sensors are available on a mobile robotic platform, some approaches fuse information from both cameras and lidars for localization within a combined image and lidar database. Oneshot (Ratz et al., 2020) customizes a network to generate descriptors from both lidar segments and their corresponding images from a camera. It estimates the sensor pose by extracting segments from a lidar scan and matching their descriptors to a database. The study demonstrates an enhanced retrieval rate when integrating visual information compared to lidar data alone. Bernreiter et al. (2021) utilizes a spherical representation to generate descriptors for associated lidar scans and images. A notable advantage of their approach is the flexibility in accommodating different camera and lidar sensor configurations during both training and querying stages.

In recent work, Lai et al. (2022) introduces a method that employs adaptive weighting on image and lidar pairs to generate descriptors through an attention network. An adaptive weight allows for different contributions from each sensor, and the results show improved retrieval rates and robustness to changing environments. LC2 (Lee et al., 2023) is an alternative method wherein 2D images and 3D lidar point clouds are both converted into 2.5D depth images. Then, a network is trained to extract global descriptors from disparity and range images. By reducing modality discrepancy, their method can perform well in very different lighting conditions across multiple missions.

2.4 Cross-modal localization

For cross-modal localization, most previous work uses a camera to localize in a lidar map. These works are the most relevant to our paper. EdgeMap (Borges et al., 2010) extracts straight lines from the point cloud map to build a 3D edge map, and applies an edge filter for the camera image. Based on a particle filter, the likelihood of each pose hypothesis is calculated through the filtered camera image and the edge map. Building upon the line extraction idea, Yu et al. (2020) extract lines from both the camera image and the lidar map. With a predicted pose from a visual odometry (VO) system, the camera pose is iteratively optimized by minimizing projection error based on 2D-3D line correspondence. They demonstrate that their method can greatly reduce drift by registering live images to the lidar map. However, these methods primarily apply to structured environments where lines and edges are prominent features.

In the context of autonomous driving, researchers have also developed localization approaches that leverage inexpensive and widely available camera sensors. Wolcott and Eustice (2014) propose a method to generate a greyscale

synthetic view from a mesh map through shading. By minimizing the normalized mutual information between live and synthetic images, this method estimates a pose within the lidar prior map. Similarly, Pascoe et al. (2015) synthesize color images from a color textured mesh and can further compensate for camera calibration changes. However, the synthetic images produced in this decade-old work were relatively limited in quality, and localization success was restricted to the vehicle's path with fixed viewing angles.

More recently, progress have been made to enhance cross-modal image-to-lidar registration methods. Approaches like Zuo et al. (2020) register dense stereo depth to a lidar map to correct for visual odometry drift. Based on semantic scene understanding, Liang et al. (2022) utilizes the point cloud map and its semantic labels to generate a prior semantic map. During runtime, semantic cues are extracted from live images to localize within the prior semantic map.

End-to-end 2D–3D registration networks such as CMR-Net (Cattaneo et al., 2019), I2DLoc (Chen et al., 2022), and I2PNet (Wang et al., 2025) have also demonstrated strong performance on autonomous driving datasets. However, these methods operate specifically in the context of camera to lidar registration and require an initial guess in the form of a relative pose prior. In contrast, our work addresses global localization without any prior pose information. As a retrieval-based method, the recent LIP-Loc (Shubodh et al., 2024) employs contrastive learning to enable global image-to-LiDAR matching and has been shown to successfully retrieve image and lidar pairs even in a zero-shot setting on autonomous driving datasets.

3 Method

3.1 Problem definition

Our goal is to estimate the position and orientation of a single live RGB camera in a prior map, $\mathcal{M} = \{\mathcal{I}_1, \mathcal{D}_1, \mathcal{I}_2, \mathcal{D}_2, ..., \mathcal{I}_n, \mathcal{D}_n\}$, where $\mathcal{I}_i, \mathcal{D}_i$ is the i-th synthesized RGB and depth image pair, that are rendered at $\mathrm{T}_{\mathrm{WM}_i}$. The relevant frames are the earth-fixed world frame W, and the map image frame M_i.

Unless specified otherwise, the position $\mathrm{p}_{\mathrm{WM}_i}$ and orientation $\mathrm{R}_{\mathrm{WM}_i}$ of the map image are expressed in world coordinates, with $\mathrm{T}_{\mathrm{WM}} \in \mathrm{SE}(3)$ as the corresponding homogeneous transform.

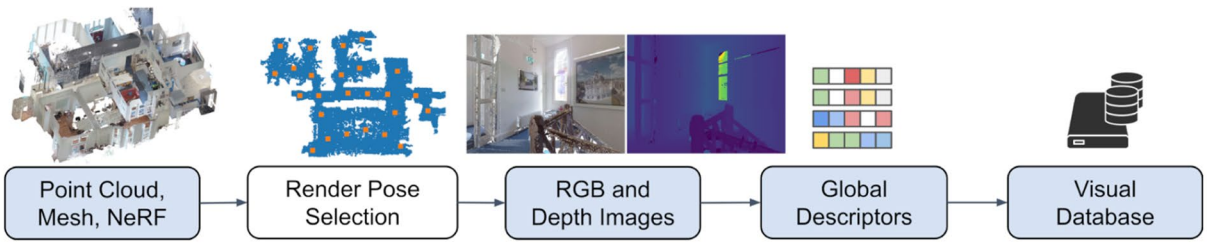
We aim to determine the pose of the live camera image $\mathcal{C}$ at a given time relative to a prior map image, defined as follows:

$$\mathrm{T}_{\mathrm{CM}_i} \triangleq [\mathrm{t}_i, \mathrm{R}_i] \in \mathrm{SO}(3) \times \mathbb{R}^3 \tag{1}$$

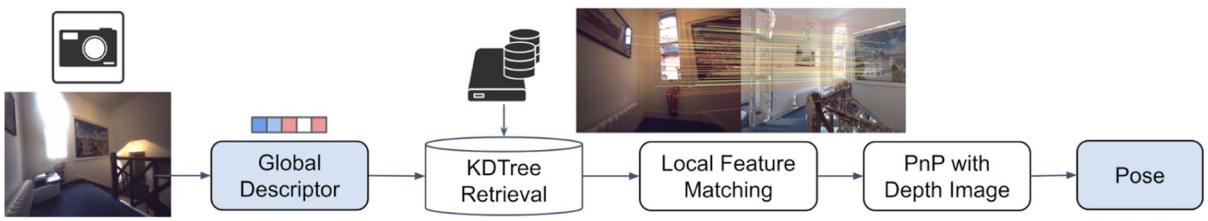
where $\mathrm{t}_i \in \mathbb{R}^3$ is the translation and $\mathrm{R}_i \in \mathrm{SO}(3)$ is the orientation of $\mathcal{C}$ relative to $\mathcal{M}_i$. Given $\mathrm{T}_{\mathrm{WM}_i}$ is known, we can find the live camera image pose in the world frame T_{WC}.

3.2 System overview

We propose a method that integrates learning-based approaches with classical visual geometry to achieve image localization within a color lidar 3D map. The system design is illustrated in Fig. 2, which shows that the system uses a color 3D map (either point cloud, mesh or NeRF based) to



(a) Step 1: Constructing a visual database from a color 3D map.



(b) Step 2: A live camera image being matched against a rendered image to obtain a relative pose estimate

Fig. 2 Overview of the system showing localization of a camera in the 3D prior map. The blue boxes represent data, and the white boxes represent algorithms

generate synthetic RGB and depth images (Step 1). Using these images, we then construct a database of global descriptors. During live operation (Step 2), the system receives a query image and generates a descriptor, which is used to retrieve the closest match from the database. Learned local features are then extracted and matched between the query and retrieved images. Using the corresponding depth image, the camera pose is subsequently estimated from the matched features and their positions in the world frame.

3.3 Pose selection when rendering a visual database

A key step is how to decide the set of poses in the 3D prior map from which the artificial images should be synthesized. The goal is to render a reasonably low number of images (to keep the map database small) while achieving coverage across the point cloud or mesh. This step is crucial because it can directly affect both the database size and the available camera views, which influences the overall localization performance. Optimal place recognition can be compromised if the virtual camera is positioned in an unsuitable location—where the map is incomplete, or the camera does not visualize during localization. Our approach involves automatically identifying free space within the map and strategically selecting rendering poses along a "free path corridor".

To address this, we propose a straightforward yet effective method that utilizes geometry and image processing to select rendering poses.

For multi-floor building scans, we begin by computing surface normals from either TLS scan data or the SLAM-generated point cloud map. As the point cloud is gravity-aligned, planar surfaces can be extracted by filtering points based on the orientation of their normals. To identify each floor, we construct a histogram of upward-facing normals as a function of point height. Once a floor is identified, its point cloud is projected onto a 2D plane to produce a top-down image. This image is then processed using a series of image processing techniques to identify reasonable walkable areas, as illustrated in Fig. 3.

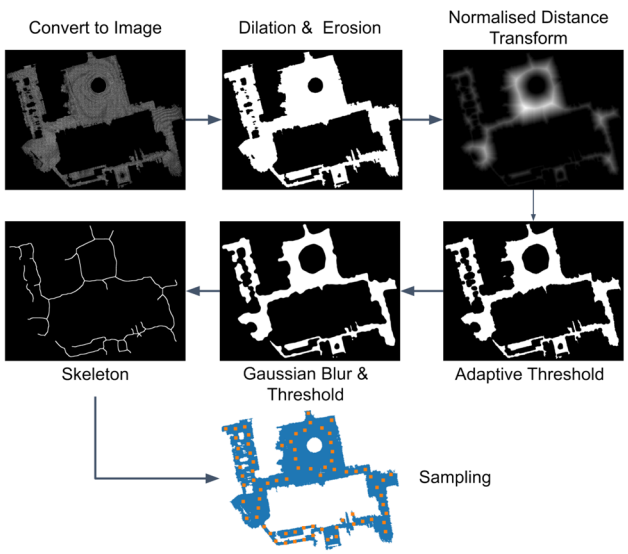


Fig. 3 Steps to establish a set of plausible rendering positions within a 3D map, which we call a "free path corridor". The map is split into floors and top-down images are rendered containing all the upward-facing points (selected using their normals). The orange points indicate the final selected positions

After extracting the floor-specific point cloud, morphological operations–dilation and erosion–are applied to fill small holes. A normalized distance transform is then computed to identify the central regions of the free space. The main free-path corridor is smoothed using adaptive thresholding followed by Gaussian blurring. A thinning operation is subsequently applied to extract a one-pixel-wide skeleton that represents the walkable paths. This skeleton is then randomly sampled at intervals of N meters to produce a set of rendering positions.

At each sampled position, four RGB/depth image pairs are generated: forward-facing, backward-facing, and two side views. This rendering pose selection method is applied uniformly across all 3D prior map representations–point clouds, meshes, and NeRFs.

3.4 Generating synthetic images

We aim to generate synthetic RGB–depth image pairs rendered from selected poses using one of three prior map representations. This section details the construction of each representation, which in turn enables the generation of synthetic images for localization. We also discuss the necessary adaptations required to improve the quality and suitability of synthetic image generation.

3.4.1 Render images from point clouds

Point clouds are point samples represented within a 3D coordinate system, sampling an object's or scene's external surface. Point clouds are commonly produced by technologies such as lidar scanning and TLS. In our study, we do not utilize point clouds derived from a SLAM system (such as FastLIO (Xu et al., 2022)), we instead use our TLS-generated point clouds as they are also used to generate precise ground truth maps with millimeter-level measurement accuracy. Further details about the TLS scanning process are provided in Sect. 4.3.

To generate RGB and depth images from the point cloud, we use the rendering capabilities of Open3D (Zhou et al., 2018), which are built upon an OpenGL framework.[2]

[2] OpenGL, Khronos Group, https://www.opengl.org/.

Fig. 4 Images from the left and right sides illustrate the before and after of rendered images from the point cloud when utilizing an adjustable point size based on the distance to the virtual camera

Several light sources are positioned around the scene to enhance specular reflections and surface detail. Virtual pinhole cameras are placed at predefined rendering poses (from the previous subsection), with their intrinsic parameters (focal lengths, principal point, and image resolution) specified for each view. The point cloud is transformed into the corresponding camera coordinate frame, and 3D points are projected onto the 2D image plane using the pinhole projection model, allowing control over the apparent pixel size of each point. For each pixel, the depth value is taken as the distance from the camera to the nearest point along the viewing ray, with pixels lacking corresponding 3D points assigned a default "no data" value. Depth maps are stored as 16-bit images in meters, scaled by a factor of 256.

One challenge we face arises when positioning the camera close to a wall: using a small fixed pixel size introduces gaps between points, resulting in a see-through effect in the rendered image, as shown on the left side of Fig. 4. To mitigate this issue, we vary the pixel size of rendered points according to an inverse-depth strategy, as defined in Eq. (2). This approach ensures that the pixel size remains within predefined minimum and maximum splatting limits. Here, $z_{(u,v)}$ denotes the depth of a point in 3D space relative to the camera frame.

$$\rho_{(u,v)} = \min \left[\max \left[\frac{\rho_{max}}{z_{(u,v)}}, \rho_{min} \right], \rho_{max} \right] \tag{2}$$

Due to a limitation imposed by OpenGL, adjusting individual point sizes dynamically is not feasible, as the parallel processing architecture requires a uniform point size for effective splatting. To circumvent this limitation, we applied Eq. (2) by rendering multiple frames, each with a point size determined by the maximum and minimum point depths

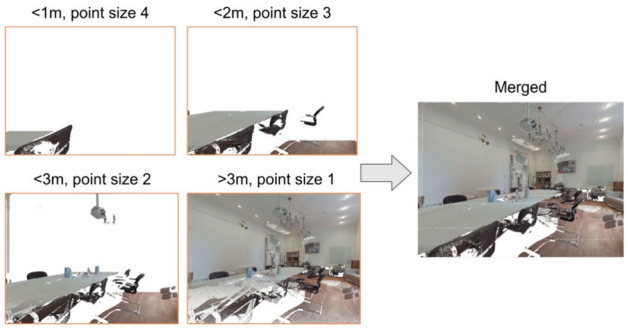


Fig. 5 Illustration of merging point cloud rendered images of different point sizes to get the final RGB and depth images

within the scene. These frames are then combined to compose the scene's final RGB and depth image. This is illustrated in Fig. 5, showcasing how four images rendered with different point sizes are merged into a single RGB image.

3.4.2 Render images from textured meshes

Meshes are widely used in robotics as dense representations for 3D scenes and objects. A mesh is a set of point vertices and the edges between them which together form a set of polygonal faces. Textured meshes are a popular method for representing static 3D scenes (Arvo, 1991; Botsch et al., 2010), as they allow surfaces to be textured with color images, accurately capturing the scene's appearance.

In our experiments, we extend our previous framework on mesh reconstruction, ImMesh (Lin et al., 2023), to reconstruct the triangular mesh of scenes captured using lidar scans. ImMesh initiates this process by estimating the pose of the lidar, followed by registering each scan to a global map. This map is partitioned into fixed-sized volumetric voxels. Mesh reconstruction is carried out online using an incremental, voxel-wise meshing algorithm. Initially, all points within a voxel are projected onto an estimated principal plane to reduce dimensionality. Subsequently, the triangular mesh is reconstructed through a series of voxel-wise operations: mesh pull, commit, and push steps. For more detailed information about the mesh construction process, readers are referred to Section VI of Lin et al. (2023). ImMesh is designed to efficiently reconstruct the mesh of large-scale scenes in real time while maintaining high geometric accuracy.

However, further modifications were necessary to color and texture the mesh reconstruction. While ImMesh constructs a mesh of a scene with adjacent triangle facets connected by edges, these facets lack color information. To address this issue, we use color images captured by the visual camera to texture the facets. The camera poses and image exposure time are estimated using R^3LIVE++ (Lin & Zhang, 2022). To enhance the smoothness and natural

appearance of the mesh texture, for each triangle facet $\mathcal{T}_i$, we blend images captured by the n nearest viewing cameras to form its texture I_i (in our experiment, n is set to 5). The pixel value $\mathrm{I}_i\,(u, v)$ at position (u, v) is blended as follows:

$$\mathrm{I}_i\,(u, v) = \frac{1}{\bar{\sigma}} \sum_{j=1}^{n} \left(\sigma_j \cdot \mathrm{I}_j\,(\boldsymbol{\pi}(\mathrm{R}_j, \mathrm{t}j, \mathrm{P})) \right) \tag{3}$$

$$\bar{\sigma} = \frac{1}{n} \sum_{j=1}^{n} \sigma_j, \quad \mathrm{P} = \mathrm{A}^{-1}(u_i, v_j) \in \mathbb{R}^3 \tag{4}$$

where $\boldsymbol{\pi}(\mathrm{R}_j, \mathrm{t}_j, \mathrm{P})$ is the camera projection function that projects a 3D point P (on the triangle facet $\mathcal{T}_i$) to the image plane using the estimated camera pose $(\mathrm{R}_j, \mathrm{t}_j)$. σ_j denotes the estimated exposure for the j-th image, and $\mathrm{A}(\cdot)$ is the wrap transform that maps P on facet $\mathcal{T}_i$ to (u, v) in texture coordinates, which can be written as $(u, v) = \mathrm{A}(\mathrm{P})$.

After reconstructing the textured mesh, we generate synthetic images of the captured scene. Similar to the point cloud, this process is accomplished by rendering the mesh into RGB and depth images using OpenGL with the predetermined rendering poses (the approach described in 3.3).

3.4.3 Render images from neural radiance fields

NeRF is an emerging 3D representation that is particularly effective for photorealistic novel view synthesis. NeRF uses an implicit representation, usually a Multi-Layer Perceptron (MLP) (Mildenhall et al., 2020), to model a radiance field. The scene is represented as a function whose inputs include a 3D location $\mathrm{x} = (x, y, z)$ and a 2D viewing direction (θ, ϕ). The output of the function is a color $\mathrm{c} = (r, g, b)$ and volume density σ.

The optimization of a NeRF is based on regulating the rendered image using the input image as the ground truth. Given the extrinsics and intrinsics of the camera, an image can be represented as a collection of rays $r(t) = o + td$. The expected color of each ray can be computed from the points sampled along the ray using volumetric integration with quadrature approximation (Max, 1995; Kajiya & Von Herzen, 1984) as $\hat{c}_u = \sum_{i=0}^{N} w_i c_i$, where:

$$w_i = \exp\left(-\sum_{j=1}^{i-1} \delta_j \sigma_j \right) (1 - \exp\left(-\delta_i \sigma_i\right)). \tag{5}$$

Early NeRF formulations were very computationally intensive—with training typically taking days until convergence. Explicit representations such as voxels (Fridovich-Keil et al., 2022; Müller et al., 2022) and 3D Gaussians (Kerbl et al., 2023) have been shown to accelerate training and rendering substantially. Nerfstudio (Tancik et al., 2023) is a well-supported open-source project which incorporates these representations. It also includes features that are effective when working with real-world data, namely scene contraction (Barron et al., 2022) which can better represent unbounded scenes, and appearance encoding (Martin-Brualla et al., 2021) which can model per-image appearance changes including lighting conditions and weather.

As with traditional visual 3D reconstruction systems such as Multi-view Stereo, it is difficult for NeRF to reconstruct textureless surfaces or locations with limited multi-view input. However, accurate depth is important for retrieval and matching, as described in Sect. 3.5. In addition, training a single NeRF for a large-scale scene is difficult due to relative small model size and limited computational hardware.

In this work we use our previous work on visual-lidar NeRF mapping called SiLVR (Scalable Lidar-Visual Reconstruction) (Tao et al., 2024). SiLVR builds upon the vision-based Nerfstudio pipeline by adding strong geometry regularization using the lidar depth measurements. With volumetric rendering, one can render both the expected depth and surface normal. SiLVR uses depth regularization (Deng et al., 2022) to encourage the ray distribution to follow a narrow normal distribution by minimizing the KL-Divergence between them:

$$\mathcal{L}_{\text{Depth}} = \sum_{\mathrm{r} \in \mathcal{R}} \mathrm{KL}[\mathcal{N}(\mathrm{D}, \hat{\sigma}) \| w(t)] \tag{6}$$

Additionally, SiLVR adds surface normal regularization to enhance the surface reconstruction. We use the same loss function as MonoSDF (Yu et al., 2022), but instead of learning-based surface normal prediction for supervision, we estimate surface normals directly using the lidar range image. The rendered surface normal is computed as the negative gradient of the density field and is supervised using the following loss function:

$$\mathcal{L}_{\text{Normal}} = \sum_{\mathrm{r} \in \mathcal{R}} \|\hat{N}(\mathrm{r}) - \bar{N}(\mathrm{r})\|_1 + \left\| 1 - \hat{N}(\mathrm{r})^\top \bar{N}(\mathrm{r}) \right\|_1 \tag{7}$$

When training the NeRF model, there are several other considerations. Training large-scale scenes (such as Blenheim Palace in Fig. 1) usually leads to inferior results compared to a smaller scene. One reason for this is the limited model size—the representation power of our learnt model is determined partially by the number of parameters that can be optimised. A relatively small model trained with a large dataset can lead to under-fitting. Additionally, loading thousands of images from a large-scale scene into RAM memory is not always practical. Therefore, SiLVR adopts a submapping system to divide the scene into smaller overlapping

submaps. Specifically, the submaps are created by applying Spectral Clustering (Von Luxburg, 2007) to divide the full global trajectory into smaller sections. The submaps overlap—with part of the global trajectory reused at the submap boundaries. This allows a smooth transition from one submap to another.

After training, a NeRF model can render both color and depth images from any desired camera viewpoint by querying its learned continuous volumetric representation. For each pixel in the target image, a ray is cast from the virtual camera into the scene, and the NeRF network predicts the volume density and color at multiple sampled points along that ray. The final pixel color is obtained by integrating these predictions, where each sample's contribution is weighted according to the volume rendering equation. The depth value is computed in a similar manner, using the same weighting coefficients but applied to the samples' distances from the camera plane.

3.5 Retrieval and matching

Upon completion of the synthetic image generation process and the subsequent creation of corresponding depth images, the resulting image database is then input into the NetVLAD (Arandjelović et al., 2018) network. This network is built with a convolutional neural network and a special VLAD layer. The central element of VLAD layer is inspired by the image representation technique known as the Vector of Locally Aggregated Descriptors. The network generates descriptors capable of accurately identifying the location of the query image, even amidst significant clutter (e.g., people, cars), variations in viewpoint, and stark differences in illumination, including day and night conditions. The descriptors of all the rendered RGB images are stored in the KDTree data structure for future indexing.

Upon receiving an incoming live camera image, our system initially undergoes a distortion correction process before being subjected to the same NetVLAD network. This step produces a descriptor for the live camera image. Afterward, the KD-Tree retrieves the virtual image that exhibits the closest match based on the descriptor values.

For each pair of matched live camera and virtual images obtained in the previous step, local features are extracted using the SuperPoint algorithm. Due to the substantial domain gap between synthetic and real images, we observe that traditional, non-learning-based feature detectors such as SIFT (Lowe, 2004) fail to identify consistent correspondences. In contrast, SuperPoint is initially pre-trained on a synthetic dataset, which enhances its domain adaptation capability. It is then further refined through multi-scale, multi-transform augmentation, enabling self-supervised training of interest point detectors. As a result, SuperPoint achieves highly repeatable feature detection across varying viewpoints.

Subsequently, the extracted local features are matched through SuperGlue (Sarlin et al., 2020). SuperGlue combines a Graph Neural Network with an Optimal Matching layer, trained specifically to establish correspondences between two sets of sparse image features.

Finally, with access to the matched local features and the corresponding depth image, the Perspective-n-Point (PnP) algorithm (Marchand et al., 2016) is used to estimate the relative pose between the live camera image and the rendered reference image. The depth image provides the 3D coordinates of matched keypoints in the reference view by back-projecting their 2D pixel locations into the 3D map, enabling the PnP solver to relate these 3D points to their 2D projections in the live image. Given the known pose of the reference image, the resulting transformation allows us to localize the live query image within the color 3D map. The pseudo-code outlining this matching and pose estimation process is presented in Algorithm 1; note that we use the confidence scores from the SuperGlue network to filter out unreliable matches.

Input: camera intrinsic f_x, f_y, c_x, c_y, depth image D, reference image pose in world frame T_{WR}
Output: T_{WQ} (query image camera pose in world frame)

for $i = 1, 2, \ldots, N$ matched keypoint **do**
  $q_u, q_v \leftarrow$ query image keypoint[i]
  $r_u, r_v \leftarrow$ reference image keypoint[i]
  depth $P_z \leftarrow D[v][u]$
  $P_x \leftarrow (r_u - cx) * z/fx$
  $P_y \leftarrow (r_v - cy) * z/fy$
  if match confidence score $\omega > 0$ **then**
    3d points $\leftarrow [P_x, P_y, P_z]$
    image points $\leftarrow [q_u, q_v]$
  end if
end for
if image points ≤ 6 **then**
  T_{QR} , success $\leftarrow$ perspective N points with (3D points, image points)
  if success **then**
    Return $T_{WQ} = T_{WR} * T_{QR}^{-1}$
  end if
end if

Algorithm 1 Query pose estimation with reference image.

3.6 Implementation details

For map construction, the mesh-based method operates in real time, requiring approximately 10–30 ms per lidar scan frame. Detailed timing results for different lidar configurations are provided in Table IV of Lin et al. (2023). For

point cloud reconstruction, we employed a tripod-mounted scanner to acquire colored point clouds, with each location requiring approximately half a day of scanning. These point clouds were also used to generate ground-truth maps and trajectories, as described in Section 4.3. For NeRF-based 3D reconstruction, each SiLVR map was trained for 150,000 iterations, with 4,096 rays sampled from the training set per iteration. Training required approximately five hours on average for all experiments, using a single NVIDIA RTX 4090 GPU.

Synthetic images for the point cloud and mesh were generated using OpenGL-based rendering, which required only a few milliseconds per image. In contrast, synthetic image generation with the NeRF method involved an MLP inference step, resulting in a longer runtime–approximately $\sim 1\,\mathrm{s}$ per 720×540 image on an NVIDIA RTX 4090 GPU.

For the localization system, the input RGB images have 720×540 pixels resolution, and we use off-the-shelf pre-trained networks. Specifically, we use a pre-trained NetVLAD backbone based on VGG16, trained on the Pitts30K (Torii et al., 2015) dataset. Additionally, the SuperGlue network we use is pre-trained on ScanNet data (Dai et al., 2017) for indoor scenes and MegaDepth data (Li & Snavely, 2018) for outdoor environments.

The localization module of the system (Step 2) is designed for real-time operation. Upon initialization, all rendered images are loaded, global descriptors are extracted for KDTree indexing, and local features are stored in memory. All algorithms are implemented on a Dell laptop with an Intel Core i7-9850 H CPU running at 2.60GHz and a Nvidia Quadro T2000 GPU with 4GB of memory. The online localization pipeline processes each input image in approximately 0.5 s, with the following upper limits for each step: global descriptor generation (200 ms), KDTree retrieval (1 ms), local feature detection (60 ms), local feature matching (100 ms), and pose estimation (60 ms). These performance metrics enable the system to comfortably operate at 1 Hz.

4 Experiment setup and hardware

4.1 Dataset

We collected two recordings at each location: one was used to build the prior map and to render the database images, and the second was used for localization testing. The recordings were collected with a walking speed of around 1 m/s. The test locations had the following distinct characteristics:

- ORI [indoor] was collected at the Oxford Robotics Institute. The ORI dataset encompasses indoor environments such as office rooms, staircases, and a kitchen. This dataset is characterized by faster camera rotation movements than the outdoor dataset with the content in the images changing quickly as the recording device moves from room to room.
- Math [outdoor, medium scale] was collected outside the Oxford Mathematics Institute in the Radcliffe Observatory Quarter on a summer's day. This dataset is characterized by substantial lighting variations, with transitions between areas of direct sunlight and shadows cast by the buildings. The presence of bushes, trees, and lawns increases the complexity of the image rendering.
- Blenheim [outdoor, large scale] was collected at Blenheim Palace, a 300-year-old country house in Woodstock, Oxfordshire. It captures walking sequences within the palace courtyard, partially extending into the palace's main hall. This dataset is challenging as images often contain significant portions of ground and sky, while the geometric symmetry of the courtyard further adds complexity.

The test recording, `ORI`, was collected across 7 office rooms spread over 2 floors with a total trajectory length of 125 m. The `Math` trajectory is 451 m long, and the `Blenheim` trajectory is 386 m long.

Again please note that the datasets and ground truth maps used here are part of the Oxford Spires Dataset mentioned in Sect. 1.

4.2 Hardware-frontier

All experimental data was collected using *Frontier* (shown in Fig. 6), a self-developed multi-sensor unit consisting of a Sevensense Alphasense multi-camera kit with 3 orthogonal cameras of 1440×1080 pixels resolution. The device contains a Hesai QT lidar with 64 beams and 104.2 ° vertical field of view. Note that the Hesai lidar is used to build the mesh and NeRF maps and generate ground truth poses.

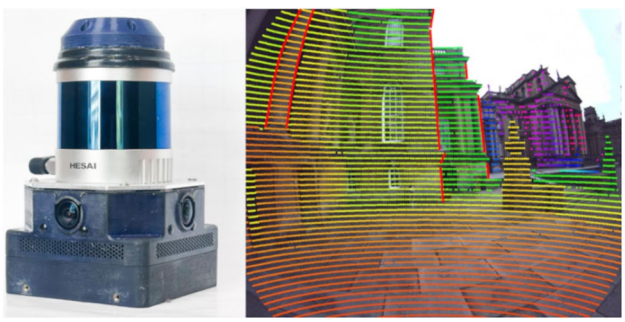


Fig. 6 Frontier multi-sensor unit consists of a 3D lidar, 3 orthogonal cameras and an IMU (left image). A fisheye image overlaid with lidar points (right image), demonstrating accurate intrinsic and extrinsic calibration (red lines are manually added to highlight the depth change)

The extrinsic calibration between the lidar and camera is established following the methodology outlined in Fu et al. (2023), ensuring sub-millimeter accuracy in translation and sub-degree precision in rotations. Accurate calibration is crucial, as shown in Fig. 6, particularly for accurately overlaying lidar scans onto color images when reconstructing color 3D maps.

4.3 Ground truth maps and trajectories

To generate ground truth point cloud maps, we used a (professional grade) Leica RTC360 Terrestrial Laser Scanner (TLS) to scan the `Math` and `Blenheim` sites and an (entry-level) Leica BLK360[3] to scan the `ORI` site. Views of these maps are shown in the left section of Table 1. TLS scans are captured around 3 m apart for indoor `ORI` and 15–20 m apart for outdoor `Math` and `Blenheim`. Note that ultra-dense distance between TLS scans is impractical due to the considerable time required to scan and process point clouds.

Each Hesai lidar scan was registered to the ground truth point cloud to obtain a 6 DoF pose. We refer readers to our previous paper, where we explain the details of the ground truth generation process in more detail (Zhang et al., 2021), which we claim achieves centimeter precision.

These point clouds were also used to render the synthetic images for the point cloud map representation, as explained in Sec 3.4.1. Before rendering, we voxel-filtered the `ORI` point cloud to 5 mm resolution and the `Math` and `Blenheim` clouds to 1 cm.

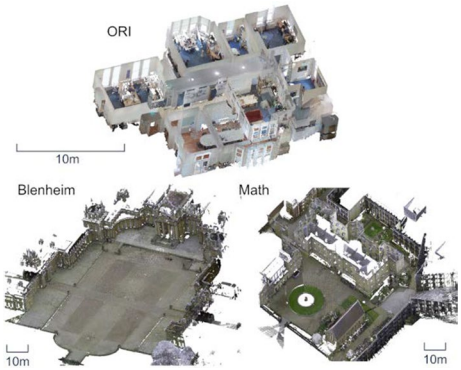
5 Experiment results

5.1 Comparison of rendered images

First we present a set of illustrative examples of rendered images obtained using the TLS point cloud, NeRF, and mesh reconstruction pipelines. Figure 7 illustrates examples drawn from the three datasets. Each pipeline has the ability to generate reasonably lifelike synthetic images. However, we note that each also exhibits distinct limitations.

For the indoor `ORI` dataset, object occlusion and the relatively sparse coverage of the TLS scanner results in some visible gaps in the point cloud, leading to white regions in the rendered images. Mesh reconstruction faces challenges in accurately representing smaller objects or intricate details, primarily due to the fixed size of the triangular faces. This method is more suitable for objects with planar surfaces than objects with complex profiles. For example, the handrails are difficult to reconstruct with a mesh, as shown in (f). Conversely, NeRF demonstrates its superiority in outdoor environments, successfully reconstructing fine details such

Table 1 Performance comparison of the different localization methods. ("DB size" is the number of images in the database. "Ret. Rate" refers to the recall@1 retrieval rate. "Loc. Rate" means the localization rate. COLMAP (*) required all the images from all three cameras to reconstruct the prior map accurately for the indoor ORI recording. "Fail" indicates that the given database images could not construct a SfM model, and therefore no query images could be localized.)



Dataset	Method		DB Size	Ret. Rate	Loc. Rate
ORI	MeshLoc		130	81	54
	HLoc	Initial	130	Fail	Fail
		Best	254	98	86
	COLMAP	Best	3020*	97	**96***
	Ours	Cloud	130	68	54
		Mesh	130	81	61
		NeRF	130	86	**67**
Math	MeshLoc		300	85	51
	HLoc	Initial	300	83	12
		Best	1019	85	39
	COLMAP	Best	1569	66	**61**
	Ours	Cloud	300	89	56
		Mesh	300	85	57
		NeRF	300	97	**75**
Blenheim	MeshLoc		210	77	55
	HLoc	Initial	210	93	42
		Best	449	94	55
	COLMAP	Best	449	85	**78**
	Ours	Cloud	210	77	57
		Mesh	210	77	56
		NeRF	210	86	**73**

[3] https://leica-geosystems.com/products/laser-scanners.

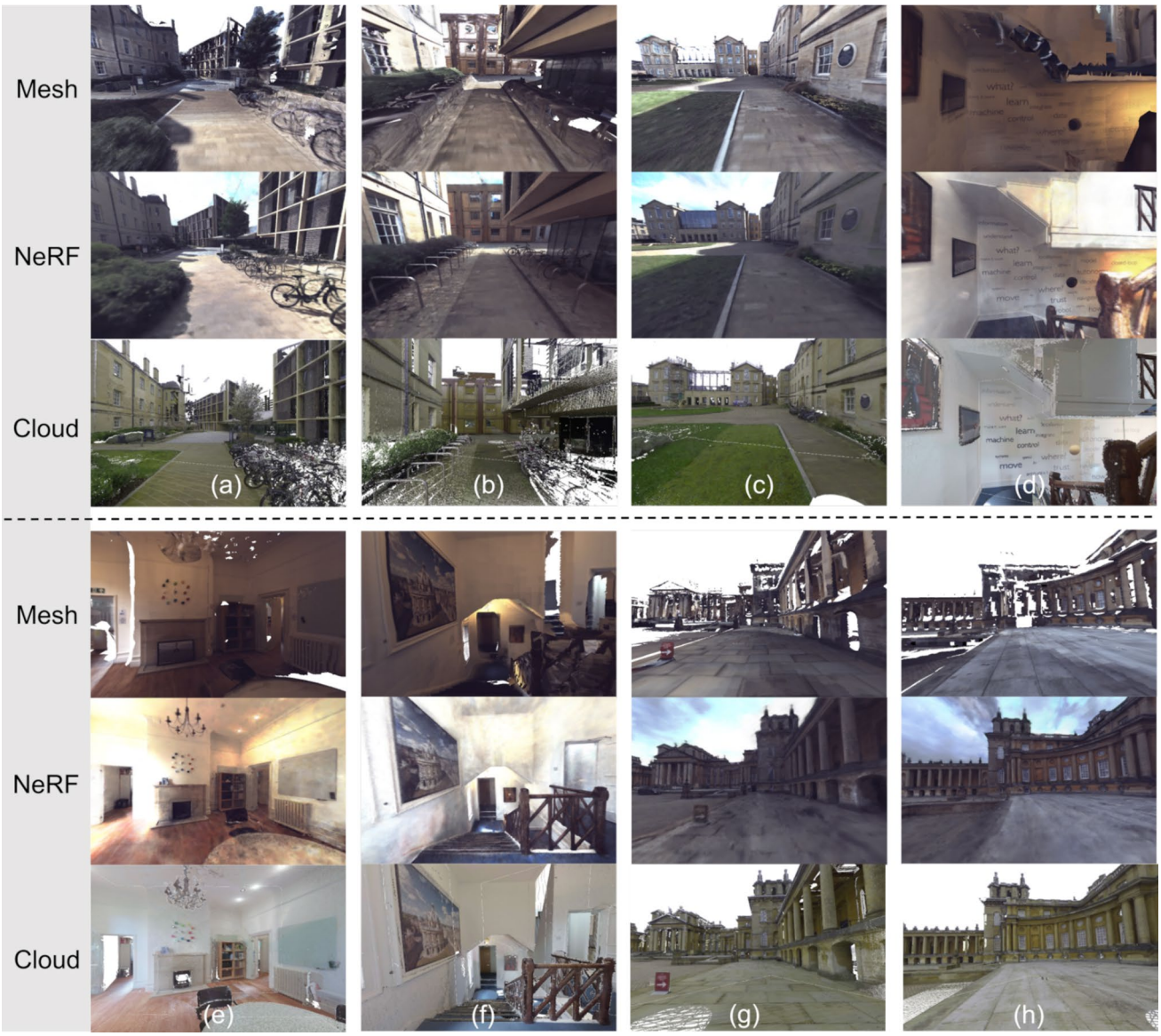


Fig. 7 A comparison of rendered images from mesh, NeRF, and point cloud in `Math` (**a**–**c**), `ORI` (**d**–**f**), and `Blenheim` (**g**, **h**)

as bicycle rails where mesh reconstruction struggles, as shown in (b). However, NeRF often fails to capture intricate ground floor patterns, as shown in (g) and (h), whereas mesh and point cloud representation, which rely on direct image projection, can readily preserve these details.

5.2 Full system evaluation

In this section, we evaluate the localization performance of the three rendering pipelines using synthesized images and benchmark them against an alternative mesh-based rendering method as well as two fully vision-based localization systems that operate on real images, as summarized in Table 1. The two vision based localization systems are HLoc and COLMAP, both of which are based on Structure-from-Motion (SfM). These methods were selected because they are widely used in the research community and are designed for large-scale, real-world reconstructions. Since both rely on real images, COLMAP serves as an approximate upper bound for achievable performance. Once the SfM reconstruction is completed, both methods enable real-time localization of query images.

Table 2 presents a high-level analysis of the suitability of these systems for key robotics tasks. The SfM approaches are non-metric, meaning their pose estimates are not scaled to real-world dimensions. This limitation necessitates other information to be available to allow effective deployment in robotic applications. The localization part is real-time, but the SfM bundle adjustment process would take a couple of hours to a day, depending on the size of the image set or

Table 2 Comparative analysis of localization methods for use in different robotics tasks. ✓ means suitable, □ means partly suitable, and ✗ means unsuitable

Method	Metric pose	Real time	Planning
HLoc COLMAP	□	✗	□
Ours-NeRF	✓	✗	✓
Ours-Cloud Ours-Mesh	✓	✓	✓

map. Comparable limitations arise for NeRF-based methods when applied to large-scale scenes, whereas mesh and point cloud maps can be constructed in real time.

In the following section, we will discuss the configurations of HLoc and COLMAP in our experiments and compare them with our methods of localizing using point cloud, mesh, and NeRF-based map representations. For a fair comparison, the primary performance metrics we use are *retrieval rate*, the proportion of images correctly retrieved as a percentage of the total number of queried images, and *localization rate*, the proportion of images correctly retrieved and localized within a threshold as a percentage of the total number of queried images. In consideration of real-time requirements for robotic operation, we use only the top-1 retrieval result for image matching and localization in all experiments. As we are evaluating global localization systems, the threshold is within 1 m and 30 ° of ground truth poses for indoors `ORI`. For outdoor `Blenheim` and `Math`, the threshold is within 2 m and 30 °.

5.2.1 Our methods–cloud, mesh and NeRF

For a fair comparison, we render an equal number of images for each dataset at the same locations. For `ORI`, render poses are 2 m apart, generating 130 images. Since `Math` and `Blenheim` cover larger areas than `ORI`, render poses are 4 m apart, and 300 and 210 images were produced, respectively.

When testing our approach of localizing with a database of rendered images, we found consistent performances for each of our three reconstruction techniques (point cloud, mesh, and NeRF) across three distinct test locations. On average, the localization rates for the point cloud and mesh approaches were 56 % and 58 %, respectively, while the NeRF-based approach achieved the best performance at 72 %. The relatively lower localization rates with point cloud and mesh systems can be attributed to the projection of color and textures from multiple camera views onto the 3D space, occasionally resulting in occluded regions and color inconsistencies.

Figure 8 illustrates the performance of pose estimation for query images localized in `Math` against a NeRF-rendered image database. The ground truth pose trajectory is drawn in blue, the estimated poses in orange dots, and the estimated poses that exceeded the threshold are highlighted in purple.

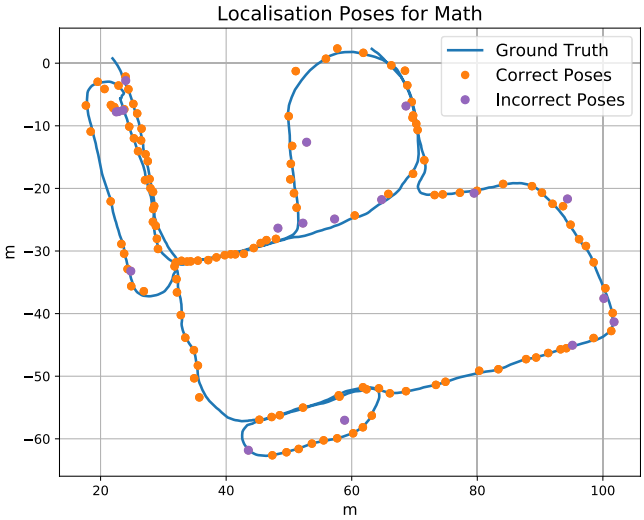


Fig. 8 Estimated poses in `Math` from the NeRF synthesized image database. The purple dots show estimated poses exceeding the 2 m and 30 ° threshold compared to the ground truth poses

A key advantage of point cloud and mesh mapping systems is their ability to generate color 3D maps online. The point cloud and mesh results are available immediately following a SLAM run, making them useful for real-time applications that cannot support rendering delays. In contrast, NeRF reconstruction offers the most photorealistic rendering outcomes but requires training after data collection. Moreover, synthetic image generation with the NeRF reconstruction is executed through an MLP inference step, which requires a relatively longer time (e.g. $\sim$ 1 s per image of resolution 720×540 on a Nvidia RTX 4090 GPU). In contrast, the OpenGL-based rendering utilized in the point cloud and mesh method takes a few milliseconds per image.

5.2.2 HLoc

HLoc (Sarlin et al., 2019) is a 6 DoF visual localization toolkit. It employs a hierarchical localization approach that integrates image retrieval and feature-matching techniques. We first configured HLoc to build an SfM model using all real images and then used it to register query images to the SfM model. For an objective comparison, we used the same pre-trained networks as used in our localization method—NetVLAD for image retrieval, SuperPoint for local feature detection, and SuperGlue for feature matching.

To ensure optimal performance, the experiments were initiated with the same number of database images as used in our system. The number of database images was then gradually increased until further performance improvements plateaued. Since HLoc uses learning-based image

feature detection and matching, it is more robust to significant changes in camera viewpoint and can successfully match image pairs across a wider range of perspectives. As a result, HLoc required fewer database images compared to COLMAP. In general, visual localization systems require a larger set of images to construct an accurate SfM model (Table 2).

To ensure a fair evaluation, we performed the following two steps for both HLoc and COLMAP. To obtain query image pose estimates (up to the scale metric), we first refine the pose estimates by eliminating any erroneous matches and then use Evo[4] to align the query image poses with the ground truth poses to recover the scaling factor and the alignment transformation. Next, we applied this scaling factor to all estimated poses and computed the localization rate.

During the initial SfM reconstruction step, HLoc performed well with indoor data but encountered difficulties in accurately constructing the larger outdoor environments of `Math` and `Blenheim`. We found that constructing a robust SfM solution requires a dense set of images with substantial overlap and minimal lighting changes. For `ORI`, an image database of 130 images (as used for map rendering pipelines) was insufficient as the images were too sparse for the SfM solver to achieve sufficient overlap. Increasing the number of images (to 254 and later 500) enabled the creation of a complete reconstruction, albeit with some observable drift in the staircases connecting the two floors of the building. For `MATH`, the best results were achieved with 1019 images, and increasing the database (to 2040 images and more) did not noticeably improve the SfM model. Lastly, for `Blenheim`, HLoc achieved its best performance with 449 images, and doubling the image count or more did not improve the overall SfM accuracy.

Overall, HLoc worked well in the indoor setting, but the reconstruction was less successful in large outdoor scenes.

5.2.3 COLMAP

This system (Schönberger & Frahm, 2016) is a widely-used SfM and Multi-View Stereo pipeline with graphical and command-line interfaces. It can construct visual maps using ordered and unordered image collections. In our experiments, we adopted the recommended approach of utilizing SIFT as the feature detector and a vocabulary tree for image retrieval. Our results show that COLMAP requires a higher number of database images than HLoc. It achieves its best performance when incorporating images from the three cameras of the Frontier (front, left, and right facing) within the dataset. This was particularly the case for the indoor `ORI` dataset, where the camera motion is more abrupt/jerky. For `ORI`, COLMAP required all images from all three cameras—facing left, forward, and right, a total of 3020 images.

Once the SfM solution is accurately estimated, localizing new images is straightforward, especially if a query image closely resembles the database images. For `MATH`, the query and map images were captured in quite different lighting conditions. Hence, the performance of image retrieval is lower with COLMAP as it uses the non-learning-based image retrieval method. Across all three locations, COLMAP gave good localization performance, but constructing the SfM solution takes double the amount of time compared to HLoc, e.g., for `MATH`, COLMAP takes around 4.5 h with 1569 images, and HLoc takes 2 h for 1019 images.

COLMAP's best performance is comparable to, or in some cases, exceeds that of our proposed method. This advantage can largely be attributed to the use of real images, as opposed to synthetically rendered ones. The comparisons with COLMAP and HLoc are intended to serve as an upper bound on performance, as examplers of mature, well-established methods that operate on real imagery rather than synthetic data.

5.2.4 MeshLoc

MeshLoc (Panek et al., 2022) is a mesh-based visual localization method that uses dense 3D meshes instead of sparse SfM point clouds. By enabling on-the-fly rendering for any feature type, it achieves state-of-the-art performance even with untextured geometry.

Based on the findings in the original MeshLoc paper, textured meshes outperform untextured meshes. Therefore, we used the textured mesh for image retrieval and matching. The open-source MeshLoc code provides the localization pipeline but requires the user to construct the mesh offline, render RGB and depth images, and supply the top retrieval results for each query image. For a like-for-like comparison, we used the same mesh and rendered images as in our method, as well as the top-1 retrieval results obtained using NetVLAD. The results are on par with our mesh-based approach, with minor differences attributable to implementation details in the localization pipeline. For instance, our method employs SuperPoint for local feature detection, whereas MeshLoc uses Patch2Pix.

5.3 Image descriptor, feature detector and matcher

In our study, we evaluated a range of image descriptors and feature detection methods, including EigenPlaces (Berton et al., 2023) and MixVPR (Ali-bey et al., 2023). We found that NetVLAD marginally outperformed the others when tested on real query images and synthetic database images. A more detailed ablation study comparing learning-based and

[4] https://github.com/MichaelGrupp/evo

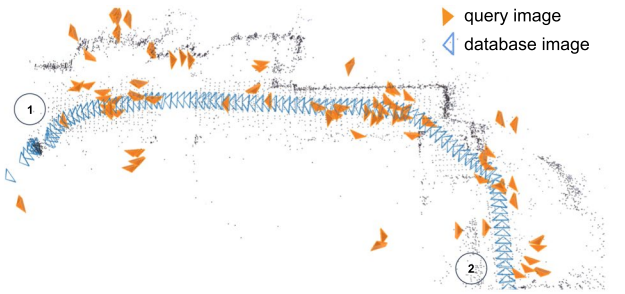
non-learning-based feature detectors and matchers is provided in Sect. 5.6. It is important to note that our approach is not dependent on the specific learning-based detector and matcher—which can be swapped for improved methods as they emerge. The key point here is that by sampling a database of synthetic images, we can leverage dense 3D maps to provide a *unified representation* across different map types. This approach offers flexibility in generating poses within the 3D map and even allows for an iterative "render and compare" strategy if needed, while also reducing the number of database images requires, as compared to SfM-based approaches.

5.4 Localization in directions unseen during mapping

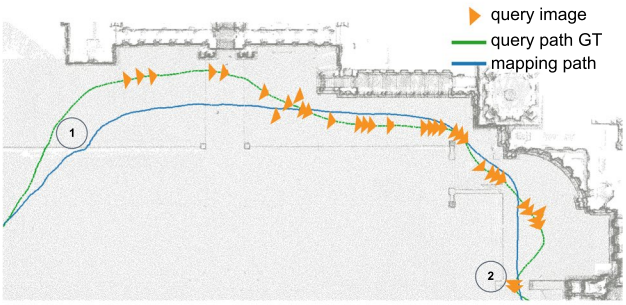
Our method has another advantage of generating images from any viewpoint. This is useful when attempting to localize when traveling in directions opposite to that used to create the map. For instance, in the scenario where the map is constructed by traversing from left to right, as shown in Fig. 9, we can synthesize images from all four orthogonal directions. This advantage facilitates localization even when traveling in the reverse direction, a challenge often encountered in conventional visual localization methodologies, which typically requires storing images from both travel directions in the database (Table 3).

The initial recording was used to reconstruct a mapping database with only the front-facing camera, as shown in Fig. 9a. The map was generated by traversing from point 1 to point 2, where 98 images were captured (illustrated in blue). We then attempted to localize to the map during a reverse traversal from points 2 to 1, with the results depicted using orange-colored camera views. Of the 87 query images, only 4 were successfully localized (when turning around the corner) due to minimal overlap in the image views. A similar pattern was observed with COLMAP where it failed to localize any query images even though the localization module had an accurate SfM reconstruction. HLoc performed marginally better than COLMAP due to its learning-based feature matching approach.

NeRF generated blurrier images from reverse perspectives when trained exclusively with front-facing camera images, resulting in suboptimal localization performance. This issue is likely caused by insufficient training views when the path involves walking in a straight line. In contrast, the mesh-based pipeline demonstrated strong performance in this scenario. Mesh could effectively generate rendered images facing in the un-mapped direction (albeit with imperfections resulting in holes in the rendered images). Despite these visual limitations, mesh-rendered images achieved a localization rate of approximately 33 %.



(a) HLoc map: Although the SfM reconstruction is accurate, only four camera views could be correctly localized around the bend.



(b) Mesh map: 30 out of 90 query images could be correctly localized.

Fig. 9 Localizing when traveling in a direction opposite to the mapping phase. The mapping direction (blue) is from 1 to 2, and the query direction (green) is from 2 to 1. Orange triangles represent the localized camera view results

Table 3 Localization performance when traveling in the opposite direction of the mapping trajectory using novel view synthesis. DB denotes "database." A larger number of cameras during mapping results in more database images

Method	DB Camera	DB Images	Ret. Rate%	Loc. Rate%
COLMAP	front	98	2	0
HLoc	front	98	7	4
Ours-NeRF	front	30	10	7
Ours-Mesh	front	30	37	**33**
COLMAP	front + side	196	29	26
HLoc	front + side	196	41	39
Ours-NeRF	front + side	30	88	**75**
Ours-Mesh	front + side	-	-	-
Ours-Point Cloud	omni	30	98	**88**

Bold values indicate the best performing method for localisation rate

In a second experiment, we expanded the mapping database to include images from two cameras, the side and front camera. As expected, this enhanced the performance for COLMAP and HLoc, with side camera views available for reverse image matching. NeRF exhibited much better localization performance, matching 75% of the images. Note that

the mesh reconstruction implementation only works with one camera.

To provide a broader comparison, we show that point clouds captured with an omnidirectional camera (both front and backward facing) can achieve a very high localization rate.

To conclude, this section demonstrates that our system can generate synthetic images in novel views, which aids the localization in a direction that is unseen during the mapping. In particular, mesh-based map representation can render images for localization even when there are very limited views of the environment, while NeRF can render better images for localization with a moderate number of views.

5.5 Ablation study: changes in environment

When using a prior map for localization, one would expect that changes to the scene would affect algorithm performance. In indoor environments, furniture and decorations are often reconfigured. Meanwhile, outdoor environments are affected by seasonal and lighting changes, while dynamic objects such as parked cars or bicycles may be added or removed. Our method must maintain a certain degree of resilience to such changes (Table 4).

To demonstrate the robustness of our approach to scene change, we tested our camera localization system with data collected 2-3 months after the initial construction of the maps. The right-hand side of Fig. 10 shows two images from `MATH` with changes (shown with red circles) in the vegetation color, the location of parked cars, and the configuration of the bicycle stand. On the left-hand side, images from `ORI` show rearranged furniture and equipment.

Table 4 Localization performance amid scene changes in prior maps

Image Source	Dataset	Time	Ret. Rate	Loc. Rate
Point Cloud	ORI	Original	68	**54**
		3 Months later	**70**	48
NeRF	Math	Original	**97**	**75**
		2 Months later	91	51
Mesh	Math	Original	**85**	**56**
		2 Months later	85	49

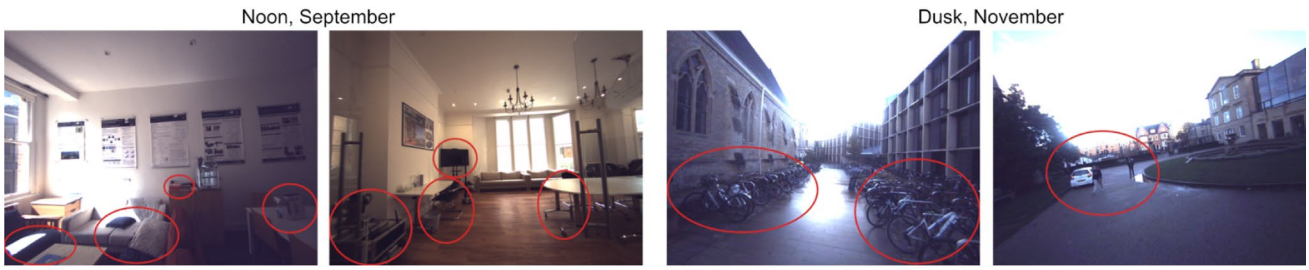
Bold values indicate experiment results with better performance

In both locations, lighting variation between the 2-3 month period was clearly observed. The detailed results of localization performance are presented in Table 4. All three map representations show a small drop in retrieval and location rates. However, they can still localize around 50% of query images, demonstrating the robustness of matching against one unified synthetic image database, aided by learning-based visual feature detector SuperPoint and matcher SuperGlue (Table 5).

5.6 Ablation study: feature detection and matching

As shown in Fig. 7, there is a significant domain gap between images captured by real cameras and synthetically rendered images. Each map representation has its own limitations

(a) Scenes in the original maps.

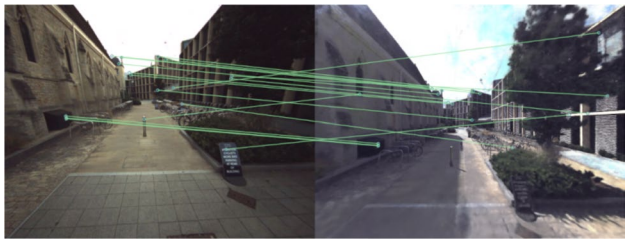


(b) Changes after three months.

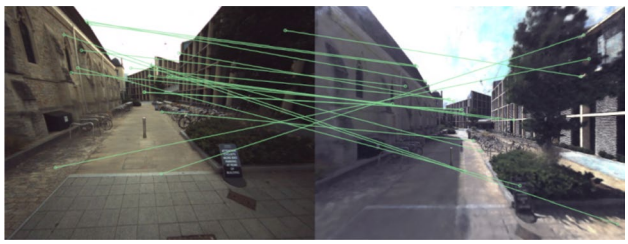
Fig. 10 Ablation study: assessing changes in `Math` and `ORI` 2-3 months apart. The top row displays images from the original dataset, while the bottom row shows corresponding images captured months later which reflect the typical movement of objects within the scene. The images on the right-half are from `Math`, while the bottom right ones were captured on a dark day. Please note that images from dust have been brightened for easier interpretation here

Table 5 Average performance of different feature detector combinations for matching a set of ten real and synthetic image pairs. An example is shown in Fig. 11. Note that many of the features matched for Akaze and SIFT are incorrect

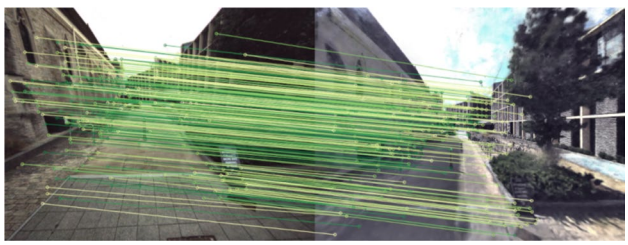
Method	Feature Detected	Matched
Akaze Alcantarilla et al. (2013) + NN	410	18
SIFT Lowe (2004) + KNN	841	21
R2D2 Revaud et al. (2019) + NN	1024	69
D2Net Dusmanu et al. (2019) + NN	2932	132
SuperPoint DeTone et al. (2018) + SuperGlue Sarlin et al. (2020)	1024	201



(a) Detector: OpenCV Akaze, Matcher: Nearest Neighbor



(b) Detector: OpenCV SIFT, Matcher: Flann KNN



(c) Detector: SuperPoint, Matcher: SuperGlue

Fig. 11 Example of matching a camera image to a NeRF rendered image with different feature detectors

when rendering the synthetic images. Images rendered from TLS point clouds often lack photorealism, exhibit a surreal quality, and have visible gaps corresponding to unscanned regions. Conversely, mesh-rendered images can contain areas with fragmented reconstruction and struggle to faithfully represent small and intricate geometric structures. For NeRF-generated images, when the rendering poses deviate significantly from the training viewpoints, a fog-like artifact often appears in the synthesized results. We aim to investigate whether learning-based feature detectors and matching techniques can help bridge the domain gap between synthetic and real images and mitigate these artifacts.

In this section, we conduct a comparative analysis of different feature detectors and matchers using a representative dataset of ten image pairs drawn from across the three datasets. One image is a live camera image, and the other is a rendered image from either the point cloud, mesh, or NeRF pipelines. Figure 11 shows one scene with common challenges: vegetation, low-textured ground, and thin bike rails. Figure 11c shows how learning-based approaches such as SuperPoint and SuperGlue can effectively identify and match numerous correct features. In contrast, traditional feature detectors like Akaze (Fig. 11a) and SIFT (Fig. 11b) perform poorly in this scenario.

Table 5 presents quantitative results for this test, comparing several learning-based and classic feature detection and matching techniques. Among the learning-based feature detectors, SuperPoint, R2D2, and D2Net show promising performance by matching many features.

However, traditional methods like SIFT and Akaze match fewer features, many of which are incorrect. Note that R2D2 produces descriptors of size 128, and D2Net generates descriptors of size 512. Since the SuperGlue matcher only has pre-trained descriptors size of 256, we instead employ a nearest-neighbor (NN) matcher with a ratio test for R2D2 and D2Net. SuperPoint and R2D2 are capped at 1024 features, while D2Net, a dense image feature extractor, naturally yields more feature points.

In summary, leveraging learning-based feature detectors and matching techniques is crucial to mitigate the domain gap between camera-captured and synthetic images, enabling robust feature detection and matching across diverse image representations.

6 Conclusions and future work

6.1 Conclusions

Using readily available 3D color maps generated from SLAM missions or TLS scanners, we can repurpose these maps for localization tasks. In this paper, we introduce a localization system specifically designed to localize a single camera image within 3D color maps by sampling and generating a synthetic image database. We first demonstrated a pipeline to construct 3D prior maps using three distinct representations: point clouds, meshes, and NeRF. Each representation is then used to synthesize RGB and depth images. Following this, we proposed a strategy to define the set of rendering poses that optimize the creation of a representative 3D map while retaining a minimal set of database images. The visual localization pipeline can estimate the

camera pose of a query image through a retrieval and matching process, leveraging learning-based descriptors and feature detectors. We conducted a comprehensive analysis of localization performance across these representations and discussed their respective merits. Additionally, we offered a benchmark comparison with two purely vision-based localization systems to situate our results within the wider field of visual localization. Notably, both point cloud and mesh representations achieve a localization accuracy of 55 % for query images, while the NeRF representation surpasses these, achieving a localization rate of 72 %.

6.2 Limitation and future work

An obvious limitation of the proposed approach lies in the quality and completeness of the prior map, as well as the coverage of the rendered synthetic images. If a region of the environment is missing from the prior map, localization will fail in that area regardless of the quality of the rendered images. Conversely, for effective scene representation, the synthetic images must comprehensively cover the areas of interest from multiple viewing angles to maximize the likelihood of successful localization.

6.2.1 Pose selection for rendering

We could consider a more sophisticated viewpoint generation method that operates directly in 3D space rather than relying on 2D image projection. Future improvements may incorporate the spatial volume of the environment, allowing viewpoints to be generated at varying heights and with adjustable camera pitch angles, enabling more diverse and comprehensive coverage.

6.2.2 Scene change

While we have studied the effect of scene changes within the context of a 3D color map, we acknowledge that there would likely be a performance decrease during gradual scene transitions over a period of months and years. In future research, we aim to devise a detection algorithm capable of promptly identifying these changes in real time, thus facilitating the identification of out-of-date regions of the map. Additionally, in scenarios where the device is equipped with a lidar sensor, we propose to update the existing map following a remapping process to ensure its continued accuracy and relevance.

6.2.3 Synthetic image

All three map representations encounter specific challenges in accurately rendering scenes with minimal texture details. This difficulty is particularly notable in 3D mapping and reconstruction, where effectively capturing the RGB texture details of ground and vegetation proves challenging. Furthermore, during the online localization of our system, images that predominantly contain low-texture objects often yield insufficient features for matching, resulting in inaccurate pose estimations. To address this, we could prune the database images during online localization to retain only the most useful images and synthesizing new ones on-the-fly as needed.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10514-025-10232-5.

Author contributions L.Z wrote the main manuscript text. Y.T and J.L. contributed in carry out experiments and evaluations. J.L. wrote Section 3.4.2 and Y.T wrote Section 3.4.2. All authors reviewed the manuscript.

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no Conflict of interest.

References

Alcantarilla, P.F., Nuevo, J., & Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces. In: British Machine Vision Conf. (BMVC). https://doi.org/10.5244/C.27.13

Ali-bey, A., Chaib-draa, B., & Giguère, P. (2023). MixVPR: Feature mixing for visual place recognition. In: IEEE/CVF Winter Conf. on Applications of Computer Vision, pp. 2998–3007.

Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2018). NetVLAD: CNN architecture for weakly supervised place recognition. *IEEE Trans. Pattern Anal. Mach. Intell., 40*(6), 1437–1451. https://doi.org/10.1109/TPAMI.2017.2711011

Arvo, J. (1991). *Graphics Gems II*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., & Hedman, P. (2022). Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR52688.2022.00539

Bernreiter, L., Ott, L., Nieto, J., Siegwart, R., & Cadena, C. (2021). Spherical multi-modal place recognition for heterogeneous sensor systems. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 1743–1750. https://doi.org/10.1109/ICRA48506.2021.9561078 . IEEE

Berton, G., Trivigno, G., Caputo, B., & Masone, C. (2023). EigenPlaces: Training viewpoint robust models for visual place recognition. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR), pp. 11080–11090. https://doi.org/10.1109/ICCV51070.2023.01017

Besl, P. J., & McKay, N. D. (1992). A method for registration of 3D shapes. *IEEE Trans. Pattern Anal. Mach. Intell., 14*(2), 239–256. https://doi.org/10.1109/34.121791

Borges, P., Zlot, R., Bosse, M., Nuske, S., & Tews, A. (2010). Vision-based localization using an edge map extracted from 3D laser range data. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 4902–4909. https://doi.org/10.1109/ROBOT.2010.5509517

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., & Lévy, B. (2010). Polygon Mesh Processing. A K Peters, Natick, Massachusetts. https://doi.org/10.1201/b10688

Cattaneo, D., Vaghi, M., Ballardini, A.L., Fontana, S., Sorrenti, D.G., & Burgard, W. (2019). Cmrnet: Camera to lidar-map registration. In: IEEE Intelligent Transportation Systems Conference (ITSC), pp. 1283–1289. https://doi.org/10.1109/ITSC.2019.8917470

Chen, X., Vizzo, I., Läbe, T., Behley, J., & Stachniss, C. (2021). Range image-based lidar localization for autonomous vehicles. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 5802–5808. https://doi.org/10.1109/ICRA48506.2021.9561335

Chen, K., Yu, H., Yang, W., Yu, L., Scherer, S., & Xia, G.-S. (2022). I2d-loc: Camera localization via image to lidar depth flow. ISPRS Journal of Photogrammetry and Remote Sensing 194, 209–221. https://doi.org/10.1016/j.isprsjprs.2022.10.009

Cummins, M., & Newman, P. (2008). FAB-MAP: Probabilistic localization and mapping in the space of appearance. *Intl. J. of Robot. Res., 27*(6), 647–665. https://doi.org/10.1177/0278364908090961

Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR.2017.261

DeTone, D., Malisiewicz, T., & Rabinovich, A. (2018). SuperPoint: Self-supervised interest point detection and description. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 224–236. https://doi.org/10.1109/CVPRW.2018.00060

Deng, K., Liu, A., Zhu, J.-Y., & Ramanan, D. (2022). Depth-supervised nerf: Fewer views and faster training for free. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR), pp. 12882–12891. https://doi.org/10.1109/CVPR52688.2022.01254

Dubé, R., Cramariuc, A., Dugas, D., Sommer, H., Dymczyk, M., Nieto, J., Siegwart, R., & Cadena, C. (2020). Segmap: Segment-based mapping and localization using data-driven descriptors. *Intl. J. of Robot. Res., 39*(2–3), 339–355. https://doi.org/10.1177/0278364919863090

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., & Sattler, T. (2019). D2-Net: A trainable cnn for joint detection and description of local features. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR.2019.00828

Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., & Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR), pp. 5501–5510. https://doi.org/10.1109/CVPR52688.2022.00542

Fu, L.F.T., Chebrolu, N., & Fallon, M. (2023). Extrinsic calibration of camera to lidar using a differentiable checkerboard model. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 1825–1831. https://doi.org/10.1109/IROS55552.2023.10341781

Galvez-López, D., & Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robotics, 28*(5), 1188–1197. https://doi.org/10.1109/TRO.2012.2197158

Gehring, C., Fankhauser, P., Isler, L., Diethelm, R., Bachmann, S., Potz, M., Gerstenberg, L., & Hutter, M. (2021). ANYmal in the Field: Solving industrial inspection of an offshore hvdc platform with a quadrupedal robot. In: Field and Service Robotics, pp. 247–260. Springer, Singapore. https://doi.org/10.1007/978-981-15-9460-1_18

Hausler, S., Garg, S., Xu, M., Milford, M., & Fischer, T. (2021). Patch-NetVLAD: Multi-scale fusion of locally-global descriptors for place recognition. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR), pp. 14141–14152 . https://doi.org/10.1109/CVPR46437.2021.01392

He, L., Wang, X., & Zhang, H. (2016). M2DP: A novel 3D point cloud descriptor and its application in loop closure detection. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 231–237. https://doi.org/10.1109/IROS.2016.7759060

Himstedt, M., Frost, J., Hellbach, S., Böhme, H.-J., & Maehle, E. (2014). Large scale place recognition in 2D lidar scans using geometrical landmark relations. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 5030–5035. https://doi.org/10.1109/IROS.2014.6943277

Kajiya, J. T., & Von Herzen, B. P. (1984). *Ray tracing volume densities. SIGGRAPH, 18*(3), 165–174. https://doi.org/10.1145/800031.808594

Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. on Graphics **42**(4). https://doi.org/10.1145/3592433

Kim, G., Kim, A. (2018). Scan Context: Egocentric spatial descriptor for place recognition within 3D point cloud map. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 4802–4809. https://doi.org/10.1109/IROS.2018.8593953

Kong, X., Yang, X., Zhai, G., Zhao, X., Zeng, X., Wang, M., Liu, Y., Li, W., & Wen, F. (2020). Semantic graph based place recognition for 3D point clouds. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 8216–8223. https://doi.org/10.1109/IROS45743.2020.9341060

Lai, H., Yin, P., & Scherer, S. (2022). AdaFusion: Visual-lidar fusion with adaptive weights for place recognition. IEEE Robot. Autom. Lett. (RA-L) 7(4), 12038–12045 https://doi.org/10.1109/LRA.2022.3210880

Lee, A.J., Song, S., Lim, H., Lee, W., & Myung, H. (2023). $(LC)^2$: Lidar-camera loop constraints for cross-modal place recognition. IEEE Robot. Autom. Lett. (RA-L) **8**(6) https://doi.org/10.1109/LRA.2023.3268848

Li, Z., & Snavely, N. (2018). Megadepth: Learning single-view depth prediction from internet photos. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR.2018.00218

Liang, S., Zhang, Y., Tian, R., Zhu, D., Yang, L., & Cao, Z. (2022). SemLoc: Accurate and robust visual localization with semantic and structural constraints from prior maps. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 4135–4141. https://doi.org/10.1109/ICRA46639.2022.9811925

Lin, J., Yuan, C., Cai, Y., Li, H., Ren, Y., Zou, Y., Hong, X., & Zhang, F. (2023). ImMesh: An immediate lidar localization and meshing framework. *IEEE Trans. Robotics*. https://doi.org/10.1109/TRO.2023.3321227

Lin, J., & Zhang, F. (2022). R^3LIVE++: a robust, real-time, radiance reconstruction package with a tightly-coupled lidar-inertial-visual state estimator. arXiv preprint arXiv:2209.03666. https://doi.org/10.48550/ARXIV.2209.03666

Liu, J., Nie, Q., Liu, Y., & Wang, C. (2023). NeRF-Loc: Visual localization with conditional neural radiance field. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 9385–9392 . https://doi.org/10.1109/ICRA48891.2023.10161420

Liu, J., Nie, Q., Liu, Y., & Wang, C. (2023). Nerf-loc: Visual localization with conditional neural radiance field. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 9385–9392. https://doi.org/10.1109/ICRA48891.2023.10161420

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Intl. J. of Computer Vision, 60*, 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Maggio, D., Abate, M., Shi, J., Mario, C., & Carlone, L. (2023). Loc-nerf: Monte carlo localization using neural radiance fields. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 4018–4025. https://doi.org/10.1109/ICRA48891.2023.10160782

Marchand, E., Uchiyama, H., & Spindler, F. (2016). Pose estimation for augmented reality: A hands-on survey. IEEE Trans. on Visualization and Computer Graphics 22(12), 2633–2651. https://doi.org/10.1109/TVCG.2015.2513408

Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., & Duckworth, D. (2021). NeRF in the Wild: Neural radiance fields for unconstrained photo collections. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR46437.2021.00713

Matsuki, H., Murai, R., Kelly, P.H.J., & Davison, A.J. (2024). Gaussian splatting SLAM. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR), pp. 18039–18048 https://doi.org/10.1109/CVPR52733.2024.01708

Max, N. (1995). Optical models for direct volume rendering. IEEE Trans. on Visualization and Computer Graphics 1(2), 99–108. https://doi.org/10.1109/2945.468400

Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., & Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In: Eur. Conf. on Computer Vision (ECCV). https://doi.org/10.1007/978-3-030-58452-8_24

Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. In: SIGGRAPH. https://doi.org/10.1145/3528223.3530127

Panek, V., Kukelova, Z., & Sattler, T. (2022). MeshLoc: Mesh-based visual localization. In: Eur. Conf. on Computer Vision (ECCV). https://doi.org/10.1007/978-3-031-20047-2_34

Pascoe, G., Maddern, W., & Newman, P. (2015). Direct visual localisation and calibration for road vehicles in changing city environments. In: IEEE Intl. Conf. on Computer Vision Workshop (ICCVW), pp. 98–105. https://doi.org/10.1109/ICCVW.2015.23

Ratz, S., Dymczyk, M., Siegwart, R.Y., & Dubé, R. (2020). OneShot Global Localization: Instant lidar-visual pose estimation. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 5415–5421. https://doi.org/10.1109/ICRA40945.2020.9197458

Revaud, J., Weinzaepfel, P., Souza, C.R., & Humenberger, M. (2019). R2D2: Repeatable and reliable detector and descriptor. In: Intl. Conf. on Neural Information Processing Systems (NeurIPS). Curran Associates Inc., Red Hook, NY, USA.

Sarlin, P.-E., Cadena, C., Siegwart, R., & Dymczyk, M. (2019). From Coarse to Fine: Robust hierarchical localization at large scale. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR) https://doi.org/10.1109/CVPR.2019.01300.

Sarlin, P.-E., DeTone, D., Malisiewicz, T., & Rabinovich, A. (2020). SuperGlue: Learning feature matching with graph neuralnetworks. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR42600.2020.00499

Sattler, T., Leibe, B., & Kobbelt, L. (2017). Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Trans. Pattern Anal. Mach. Intell., 39*(9), 1744–1756. https://doi.org/10.1109/TPAMI.2016.2611662

Schönberger, J.L., & Frahm, J.-M. (2016). Structure-from-motion revisited. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR) . https://doi.org/10.1109/CVPR.2016.445

Shubodh, S., Omama, M., Zaidi, H., Parihar, U.S., & Krishna, M. (2024). Lip-loc: Lidar image pretraining for cross-modal localization. In: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, pp. 948–957.

Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., & Ahuja, A., etal.: (2023). Nerfstudio: A modular framework for neural radiance field development. In: SIGGRAPH. https://doi.org/10.1145/3588432.3591516

Tao, Y., Muñoz-Bañón, M.Á., Zhang, L., Wang, J., Fu, L.F.T., & Fallon, M. (2025). The oxford spires dataset: Benchmarking large-scale lidar-visual localisation, reconstruction and radiance field methods. Intl. J. of Robot. Res.

Tao, Y., Bhalgat, Y., Fu, L.F.T., Mattamala, M., Chebrolu, N., & Fallon, M. (2024). SiLVR: Scalable lidar-visual reconstruction with neural radiance fields for robotic inspection. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA57147.2024.10611278

Torii, A., Sivic, J., Okutomi, M., & Pajdla, T. (2015). Visual place recognition with repetitive structures. *IEEE Trans. Pattern Anal. Mach. Intell.* https://doi.org/10.1109/TPAMI.2015.2409868

Trivigno, G., Masone, C., Caputo, B., & Sattler, T. (2024). The Unreasonable Effectiveness of Pre-Trained Features for Camera Pose Refinement https://arxiv.org/abs/2404.10438

Uy, M.A., & Lee, G.H. (2018). PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In: IEEE Intl. Conf. Computer Vision and Pattern Recognition (CVPR) . https://doi.org/10.1109/CVPR.2018.00470

Vidanapathirana, K., Moghadam, P., Harwood, B., Zhao, M., Sridharan, S., & Fookes, C. (2020). Locus: Lidar-based place recognition using spatiotemporal higher-order pooling. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 5075–5081. https://doi.org/10.1109/ICRA48506.2021.9560915

Vidanapathirana, K., Ramezani, M., Moghadam, P., Sridharan, S., & Fookes, C. (2022). LoGG3D-Net: Locally guided global descriptor learning for 3D place recognition. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 2215–2221. https://doi.org/10.1109/ICRA46639.2022.9811753

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics Comput, 17*, 395–416. https://doi.org/10.1007/S11222-007-9033-Z

Wang, G., Zheng, Y., Wu, Y., Guo, Y., Liu, Z., Zhu, Y., Burgard, W., & Wang, H. (2025). End-to-end 2d–3d registration between image and lidar point cloud for vehicle localization. *IEEE Transactions on Robotics, 41*, 4643–4662. https://doi.org/10.1109/TRO.2025.3588454

Wolcott, R.W., & Eustice, R.M. (2014). Visual localization within lidar maps for automated urban driving. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 176–183. https://doi.org/10.1109/IROS.2014.6942558

Xu, W., Cai, Y., He, D., Lin, J., & Zhang, F. (2022). FAST-LIO2: Fast direct lidar-inertial odometry. *IEEE Trans. Robotics, 38*(4), 2053–2073. https://doi.org/10.1109/TRO.2022.3141876

Yin, H., Lin, Z., & Yeoh, J. K. W. (2023). Semantic localization on BIM-generated maps using a 3D lidar sensor. *Automation in Construction, 146*, Article 104641. https://doi.org/10.1016/j.autcon.2022.104641

Yu, H., Zhen, W., Yang, W., Zhang, J., & Scherer, S. (2020). Monocular camera localization in prior lidar maps with 2D-3D line correspondences. IEEE/RSJ Intl. Conf. on Intelligent Robots and

Systems (IROS), 4588–4594. https://doi.org/10.1109/IROS45743.2020.9341690
Yu, Z., Peng, S., Niemeyer, M., Sattler, T., & Geiger, A. (2022). MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. Intl. Conf. on Neural Information Processing Systems (NeurIPS).
Zeisl, B., Sattler, T., & Pollefeys, M. (2015). Camera pose voting for large-scale image-based localization. In: Intl. Conf. on Computer Vision (ICCV), pp. 2704–2712. https://doi.org/10.1109/ICCV.2015.310
Zhang, Z., Sattler, T., & Scaramuzza, D. (2021). Reference pose generation for long-term visual localization via learned features and view synthesis. ijcv 129(4), 821–844. https://doi.org/10.1007/s11263-020-01399-8
Zhang, L., Tejaswi Digumarti, S., Tinchev, G., & Fallon, M. (2023). InstaLoc: One-shot global lidar localisation in indoor environments through instance learning. In: Robotics: Science and Systems (RSS). https://doi.org/10.15607/RSS.2023.XIX.070
Zhang, L., Camurri, M., Wisth, D., & Fallon, M. (2021). Multi-Camera LiDAR Inertial Extension to the Newer College Dataset. arXiv:2112.08854.
Zhou, Q.-Y., Park, J., & Koltun, V. (2018). Open3D: A modern library for 3D data processing. arXiv:1801.09847. https://doi.org/10.48550/ARXIV.1801.09847
Zuo, X., Ye, W., Yang, Y., Zheng, R., Vidal-Calleja, T., Huang, G., & Liu, Y. (2020). Multimodal localization: Stereo over LiDAR map. *J. Field Robot., 37*(6), 1003–1026. https://doi.org/10.1002/ROB.21936

Lintong Zhang received an M.Eng. degree in Electrical and Electronics Engineering from Imperial College London, UK, in 2015. From 2015–2020, he worked as a robotic engineer at Dyson and Oxa on applications such as domestic robots and autonomous vehicles. He gained the DPhil degree in Engineering Science from the University of Oxford, UK in 2024. His research interests include probabilistic and machine-learning methods for localization, mapping, and scene understanding.

Yifu Tao received an M.Eng. degree in Engineering Science from the University of Oxford, UK, in 2020. And gained his DPhil degree in Engineering Science from the University of Oxford, UK, in 2025. His research interests include 3D reconstruction using visual andlidar sensors and deep learning methods

Jiarong Lin received a B.S. degree in Optical Information Science and Technology from the University of Electronic Science and Technology of China (UESTC) in 2015. He received his Ph.D. degree in the Department of Mechanical Engineering, the University of HongKong (HKU), Hong Kong, China, in 2023. Currently, Jiarong Lin is with the Institute of Unmanned Systems, Beihang University, Beijing, China. His research interests include light detection, and ranging (lidar) mapping and sensor fusion.

Fu Zhang received the B.E. degree in automation from the University of Science and Technology of China (USTC), Hefei, China, in 2011 and the Ph.D. degree in controls from the University of California at Berkeley, Berkeley, CA, USA, in 2015. He joined the Department of Mechanical Engineering, University of Hong Kong (HKU), Hong Kong, as an Assistant Professor in 2018. His current research interests include robotics and controls, with a focus on UAV design, navigation, control, and LiDAR-based SLAM.

Maurice Fallon received the B.Eng. degree in electrical engineering from University College Dublin, Dublin, Ireland, in 2004 and the Ph.D. degree in acoustic source tracking from the University of Cambridge, Cambridge, U.K., in 2008. From 2008 to 2012, he was a Postdoc and a Research Scientist with MIT Marine Robotics Group working on SLAM. Later, he was the Perception Lead of MIT's team in the DARPA Robotics Challenge. Since 2017, he has been a Royal Society University Research Fellow and an Associate Professor with the University of Oxford, Oxford, U.K. He leads the Dynamic Robot Systems Group, Oxford Robotics Institute. His research interests include probabilistic methods for localization, mapping, multisensor fusion, and robot navigation.