EURASIP Journal on Advances in Signal Processing

RESEARCH **Open Access**

Robust point cloud registration for map-based autonomous robot navigation

Amit Efraim[1*] and Joseph M. Francos[1]

*Correspondence: efraimam@post.bgu.ac.il

[1] Electrical and Computer Engineering Department, Ben-Gurion University, Beer-Sheva, Israel

Abstract
Autonomous navigation in large-scale and complex environments in the absence of a GPS signal is a fundamental challenge encountered in a variety of applications. Since 3-D scans provide inherent robustness to ambient illumination changes and the type of the surface texture, we present Point Cloud Map-based Navigation (PCMN), a robust robot navigation system, based exclusively on 3-D point cloud registration between an acquired observation and a stored reference map. It provides a drift-free navigation solution, equipped with a failed registration detection capability. The backbone of the navigation system is a robust point cloud registration method, of the acquired observation to the stored reference map. The proposed registration algorithm follows a hypotheses generation and evaluation paradigm, where multiple statistically independent hypotheses are generated from local neighborhoods of putative matching points. Then, hypotheses are evaluated using a multiple consensus analysis that integrates evaluation of the point cloud feature correlation and a consensus test on the Special Euclidean Group SE(3) based on independent hypothesized estimates. The proposed PCMN is shown to achieve significantly better performance than state-of-the-art methods, both in terms of place recognition recall and localization accuracy, achieving submesh resolution accuracy, both for indoor and outdoor settings.

Keywords: Point clouds, Navigation, Registration, Place recognition

1 Introduction

Autonomous navigation in large-scale and complex environments in the absence of a GPS signal is a fundamental challenge in applications ranging from autonomous flight to autonomous driving to indoor robot navigation. Given a sequence of observations, each in the form of a 3-D scan (LiDAR, RGB-D camera) and a 3-D reference map, the problem addressed in this paper is to continuously estimate the 6-DOF of the autonomous robot relative to the initially unknown world coordinate system. The advantage of using 3-D scans for robot navigation over image-based navigation is the sensitivity of the latter, and hence, performance degradation in the presence of strong illumination and appearance variations due to lighting, weather and seasonal changes, as well as the difficulties in navigating in textureless environments. On the other hand, 3-D scans provide inherent robustness to ambient illumination or the type of the surface texture.

In this paper, we present Point Cloud Map-based Navigation (PCMN), a robust robot navigation system, based solely on 3-D point cloud registration between the acquired observation and a stored reference map. In principle, such a system is implemented through different hierarchies of solutions to the point cloud registration problem, each adapted to a different stage of the navigation process: The first stage is the initialization stage where the acquired point cloud, e.g., LiDAR frame, is registered to a stored reference 3-D map in order to determine the robot initial location. Different variants of this step are known in the literature under various names such as *place recognition* [1–3] or frame to reference map registration [4]. The second stage, the tracking stage, handles a computationally less demanding problem: the sequential registration of a stream of partially overlapping consecutive frames so that the robot 6 DOF are continuously updated starting from the initial location estimate. The tracking stage has similarities with SLAM-based odometry where frame-to-frame registration of the observations stream is employed to track the robot 6-DOF. However, frame-to-frame registration (similar to any dead-reckoning system) drifts with time, and therefore, many SLAM solutions implement a loop closure stage in order to minimize the drift by revisiting the same location. The approach described in this paper is very different as the tracking stage is implemented via registration of every acquired frame to the pre-loaded reference map and not to previous frames of a map built online during the robot motion. Thus, a key advantage of the proposed approach is that registration at every instant is implemented against an accurate reference map and hence errors are not accumulated and the drift problem of dead-reckoning systems is eliminated altogether. A closely related approach for designing the general architecture of a point cloud-based navigation system is LOL, [3]. However, while LOL employs place recognition to correct the accumulated odometry drift, the proposed PCMN does not accumulate any drift. Also, the implementation of LOL is fundamentally different in its principles from the one suggested in the current paper.

In general, the place recognition problem is defined as follows. Given a query point cloud and a reference map (often implemented as a collection of previously obtained scans), the place recognition problem is to find the location of the query in the reference map. Commonly, existing loop closure and place recognition algorithms treat this problem as a detection problem, matching the query to a collection of stored scans. As such, existing place recognition algorithms do not provide an accurate location of the vehicle (e.g., up to tens of meters in the case of vehicle mounted LiDAR), and further registration is required to obtain an accurate location. In addition, since place recognition algorithms and datasets often aim for the loop closure problem, both query and reference scans are obtained using the same modality, or the same stream, in the case of SLAM. This approach is appropriate for loop detection, but not necessarily for matching a scan to an existing reference map due to differences between the modality employed to obtain the observations and the modalities employed to generate the reference map, as well as differences in sampling patterns.

The backbone of the proposed PCMN is a registration solution that *jointly* solves both the place recognition and tracking problems relative to a pre-existing reference map. It is based on a novel point cloud registration procedure that combines the Rigid Transformation Universal Manifold Embedding framework [4, 5] for generating multiple hypotheses for the registration parameters, and a novel multiple consensus analysis, incorporating point cloud features correlation [6], with independent estimates SE(3) consensus and point correspondences consistency verification. Since the location detection problem is solved jointly with the registration problem, the proposed place recognition solution provides accuracy which is an order of magnitude better than existing algorithms, without requiring further registration. Very importantly, it incorporates a failed registration detection capability. In the place recognition task, this capability is used as an extremely accurate detector of overlap between a query and reference scans. During tracking, it is used for verification and hence for controlling possible errors in the navigation process.

The major contributions of the paper are:

- Introduction of a drift-free autonomous navigation solution—the Point Cloud Map-based Navigation (PCMN). PCMN is based on a novel method for robust point cloud registration between the observed point clouds and a reference map. PCMN achieves better than the sampling-rate accuracy.
- Introduction of a robust registration backbone, equipped with a Success/Failure registration detection capability, based on multiple consensus analysis.
- Optimizing the structure of the rigid transformation universal manifold embedding (RTUME), [7], in the framework of hypotheses generation and evaluation for point cloud to reference map registration.
- Integration of the registration backbone into a place recognition module, yielding improved performance over existing solutions when evaluated on loop closure benchmarks.

The structure of this paper is as follows: In Sect. 2, a high-level description of the navigation system architecture is presented: The design principles of its two components, the place recognition module and the tracking module, are detailed. Both modules are implemented using different configurations of the registration backbone described in Sect. 3. Section 3 elaborates on the design of the registration backbone, including the Rigid Transformation Universal Manifold Embedding for hypotheses generation and the multiple consensus analysis for hypothesis evaluation and failed registration detection. In Sect. 4, the proposed algorithm is deployed in several map-based outdoor and indoor place recognition and navigation applications, as well as in a loop closure setting. Finally, in Sect. 5 we discuss our conclusions.

2 Navigation system architecture

In this section, the overall structure of the proposed Point Cloud Map-based Navigation is presented. The proposed navigation system is composed of two components. A precision localization module, designed to estimate the location of an observed point cloud within a large reference map and a tracking module, designed to accurately track a sequence of observed point clouds with respect to the reference map, once the initial location is known. Figure 1 illustrates the application of the two components using LiDAR scans from the KITTI data set [8].

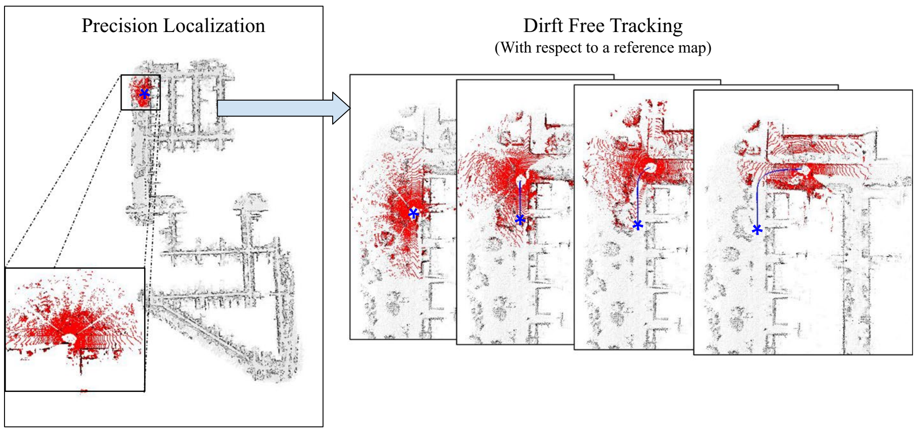


Fig. 1 Overall structure of the proposed navigation system. First, the vehicle's location is accurately estimated within the larger reference map using the proposed registration and localization backbone. (Observation is depicted in red.) Then, once the initial location is known (marked by an asterisk), drift-free tracking is performed by aligning the sequence of incoming frames to the reference map using a local registration algorithm, as seen on the right

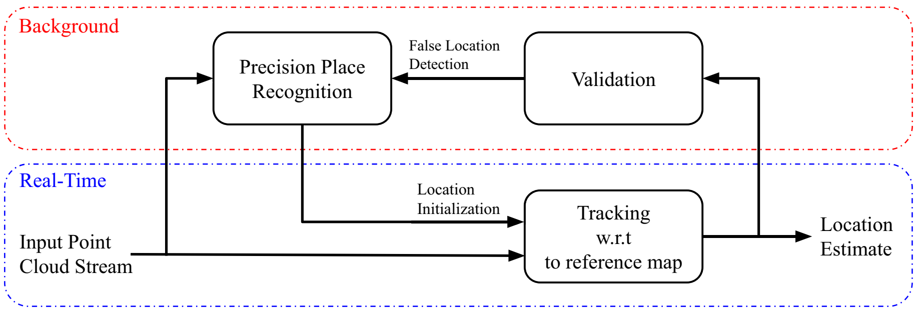


Fig. 2 Point Cloud Map-based Navigation (PCMN) block diagram. First, initialization is performed using precision place recognition (Sect. 2.1). Then, tracking is performed using local registration with respect to a reference map. Validation is performed in the background using the proposed registration backbone multiple consensus analysis (Sect. 3.2). If validation detects a false location, place recognition is initialized to recover the correct location

The precision localization module (detailed in Sect. 2.1) is based on a high-precision, robust registration pipeline, that also provides an estimate of the validity of the registration result (detailed in Sect. 3.2). The ability to detect a failed registration is used to reject false candidate locations during precision place recognition. During tracking, this ability is used to verify that tracking has not failed by validating the location using a background task, operating at predetermined intervals. Figure 2 illustrates a high-level block diagram of the proposed navigation system. The combination of precision place recognition and drift-free tracking with background validation provides a robust and accurate navigation system.

2.1 Precision localization module

The most challenging component in the design of a navigation system, in the absence of a GPS signal, is the initial localization, since it should provide a solution to the problem of registering an observed query to a reference map, using no *a priori* knowledge of the robot pose while facing all the challenges due to the differences in the properties of the observation and the reference map.

Standard registration algorithms (e.g., [9–13]) are not designed to handle registration between an observed point cloud and a much larger reference, while standard place recognition algorithms are designed as segment "detectors" and thus only provide a coarse location estimate [1, 14–16]. Thus, in order to accurately align the observed point cloud to a large reference map, two distinct steps are required. First, a coarse place recognition is performed, estimating the approximate location of the observed query (usually with precision of 10-30 meters for outdoor applications). Then, an accurate location is obtained by performing registration between the observed query and the reference map, initialized by the estimated location provided using the coarse place recognition algorithm. Place recognition has been widely explored in the imaging domain, and more recently also for 3-D point clouds. State-of-the-art methods for place recognition in 3-D share a common paradigm of comparing a query scan to a dictionary of reference map segments, or reference scans. The implementation is by attaching to each submap a global descriptor. Given a query observation, its global descriptor is evaluated and then compared against all submap descriptors in the reference "database." The query scan location is determined to be that of its nearest neighbor descriptor in the database. The pioneering method in using a learning-based method for 3-D point cloud place recognition is PointNetVLAD [16]. It employs a PointNet [17] architecture to extract local features for each submap, and then, a NetVLAD [18] layer is used to aggregate them into a global descriptor. In addition, [16] also defines a benchmark for 3-D point cloud place recognition based on the Oxford Robotcar dataset [19]. A successful place recognition for a given query is declared if its closest descriptor from the database belongs to a submap with a distance smaller than a predefined threshold from the ground truth (25 m for KITTI [8], and similar datasets).

More recent place recognition methods try to overcome the weaknesses of PointNet architecture to generalize for complex scenes by using different mechanisms to capture local contextual information. PCAN [20] extends the PointNetVLAD architecture by using sampling and grouping blocks at different scales. These are combined to the NetVLAD original block to extend the submap descriptor contextual information. LPD-Net [21] suggests enhancing the computed local features by using pre-computed handcrafted features. These are fed to a PointNet architecture together with the points' coordinates. The descriptor computed by the PointNet block is forwarded to a graph neural network that processes the neighborhood information. Finally, a global descriptor is computed by aggregating local descriptors using a NetVLAD block. MinkLoc3D [1] uses a similar approach but offers a different architecture. It employs sparse 3-D convolution to create a 3-D Feature Pyramid Network (FPN) that extracts both local and global information.

Another family of approaches is modality specific, aiming at LiDAR-carrying vehicles. These approaches achieve yaw axis invariance by using radial information in the descriptors. Scan Context [14] constructs a histogram image descriptor from the elevation and yaw angle of points. Thus, yaw axis rotation is equivalent to a shift in the descriptor. Consequently, the similarity between descriptors is found by maximizing the cross correlation between descriptor images with respect to circular shift of the descriptors. Similarly, OverlapTransformer [15] first converts the LiDAR scans into range images, to learn a descriptor from the range image, conserving the circular shift relation in the descriptor as a result of yaw axis rotation of the point cloud.

There are two main branches to the place recognition problem. Most of the existing solutions address the problem for the case where the observation and reference map are generated by the same modality, as is the case in SLAM loop closure [14], or when different scans of the same route are employed, [2, 15]. However, the more general (and difficult) problem is that of place recognition when the reference map is acquired by a different modality than the observation (e.g., GoogleMaps, or commercially available DSM).

In order to be useful for autonomous navigation of robots such as autonomous cars or indoor vehicles, the proposed method should provide zero false initializations, as well as maximal localization error in the order of 10 cm for cars, and 1 cm for indoor robots. Currently, state-of-the-art outdoors place recognition methods are able to detect the location of an observed point cloud within 25 meters of its actual location only in 90% of the attempts [15, 22]. Thus, even assuming the 25 meters initialization threshold is enough to initialize the registration step, registration will fail without warning in at least 10% of the attempts, regardless of the registration method used. Thus, the performance of the employed place recognition algorithm becomes the limiting factor of accurate localization performance.

Hence, an entirely different approach, based on a high-precision point cloud registration backbone, equipped with a mechanism to accurately detect failed registration attempts, where the place recognition and registration are performed *jointly* is suggested in this paper for implementing the precision navigation system. By rejecting reference candidates where registration failed with extremely low false-positive rates, localization success rates are increased significantly, achieving the desired performance metrics.

In the proposed registration backbone described in Sect. 3, the reference model is stored as a database of slightly overlapping segments, such that in the place recognition process, the query observation is tested for its match with every segment of the database using a high accuracy point cloud registration procedure, equipped with a success/failed registration detection capability. The database is designed such that (since the expected dimensions of a single observation are known) any observation will have a significant partial overlap with at least one segment. Figure 3 depicts an example reference map, partitioned into segments as well as a query scan registered to the reference map. The example is based on the 3DMatch dataset [23].

Due to the design of the proposed registration solution when performing registration between partially overlapping point clouds (i.e., matching query and reference segment), successful registration is obtained with a very high probability (see Table 1). Moreover, when considering non-overlapping point clouds, registration fails with a very high probability. Thus, in the place recognition module, instead of deciding based on distances between the descriptors assigned to each point cloud, a success/failed registration decision is employed to verify, with extremely high precision and low false-positive rates, putative registrations between overlapping point clouds. Furthermore, if there are no segments that result in successful registration with the query observation, the *localization* is determined to be a failure, giving the navigation system a cue to initiate recovery. The precision and false-positive rates when using

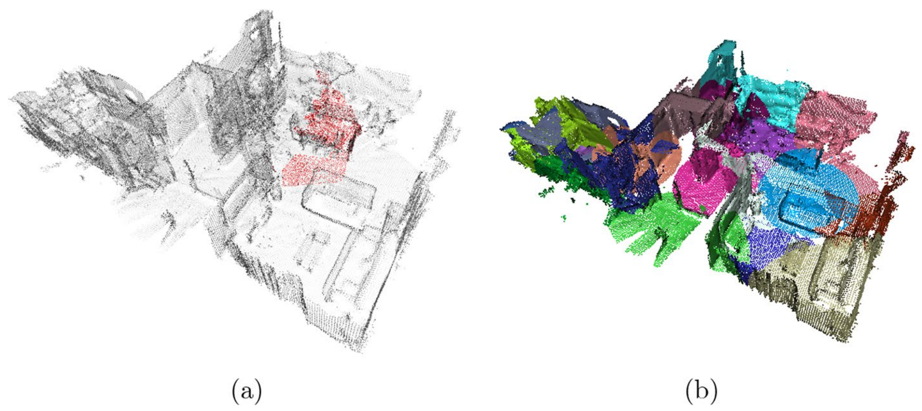


Fig. 3 **a** An example reference map from the 3DMatch dataset, with a query fragment imposed colored in red, constructed using the raw 3DMatch data. The reference map and query fragment were created using disjoint scans from the original sequence. **b** The reference map is partitioned into smaller, slightly overlapping segments (each segment is depicted using a different color)

Table 1 P_D and P_{FA} on the KITTI Place Recognition dataset

	P_D	P_{FA}
KITTI Sequence 8	0.97	0.00
KITTI Sequence 9	0.99	0.01
KITTI Sequence 10	0.98	0.0
All	0.98	0.003

the registration success/failure detection to detect partially overlapping point clouds, listed in Table 1, were evaluated on the KITTI place recognition dataset, as detailed in Sect. 4.2. In the table, P_D is defined as the probability of correctly estimating that the registration was successful when attempting registration between partially overlapping point clouds. P_{FA} on the other hand is defined as the probability of estimating that the registration was successful, while, in fact, the point clouds have no overlap.

A direct implementation of the place recognition by attempting the registration of the observation to *every* segment of the reference map is computationally prohibitive. Hence, in order to reduce the computational requirements, an existing place recognition algorithm is employed (e.g., [14, 15]) for sorting the reference segments with respect to the query point cloud, using a simple greedy algorithm, summarized in Fig. 4, and described next.

Given a query point cloud $\mathcal{Q}$, and sorted submaps from the reference map $\{\mathcal{P}_{s(i)}\}_{i=1}^{N}$, ranked using an existing place recognition algorithm, [14, 15] according to their matching score with the query point cloud, the proposed greedy algorithm consecutively attempts registration between the query point cloud and each reference segment. Once a registration attempt succeeds, the algorithm returns the estimated registration parameters and matching segment. If no segment is found to match the query it is determined that the localization failed, thus events of false place recognition are eliminated with high probability.

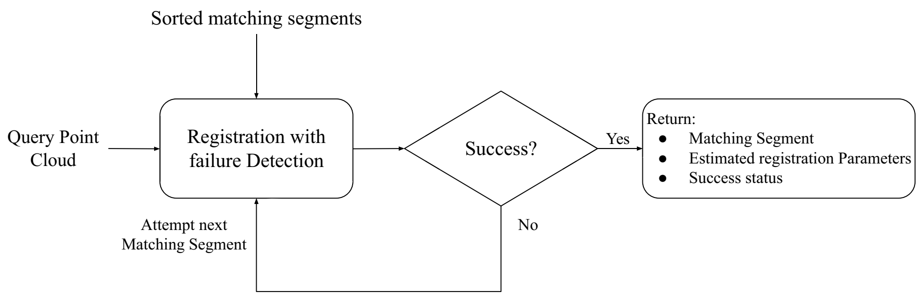

Sorted matching segments
Query Point Cloud
Registration with failure Detection
Success?
Yes
Return:
● Matching Segment
● Estimated registration Parameters
● Success status
No
Attempt next Matching Segment

Fig. 4 The proposed precision place recognition greedy algorithm. The algorithm serially attempts registration between the putative matching segments and the query point cloud. Once a successful registration attempt is made, the algorithm returns the matching candidate, registration parameters, and a success state. If no successful registration is found, the algorithm returns a failure state

2.2 Tracking module

Both in the place recognition module, as well as in the tracking module, consecutive observations are registered and aggregated into a larger point cloud fragment in order to minimize the effect of the non-regular and sparse sampling. The newly acquired fragments are registered to the reference map, thus eliminating altogether the drift problem of a dead-reckoning tracking, as tracking is not relative to the previous frame, but to a global reference. Since the approximate pose of the robot is already known based on its previous location, the registration to the global reference map is computationally simpler as the uncertainty of the pose is much smaller than in the initialization phase. In the performed experiments presented in Sect. 4, trimmed ICP [24] a robust implementation of ICP [25] is used, as well as the feature correlation matched manifold detector [26], where both methods are evaluated for their tracking accuracy with respect to a reference map.

3 The registration backbone

When the initial pose of the observed point cloud, relative to a target point cloud, is unknown, the most common approach for point cloud registration begins with the process of matching key points using local geometric features (e.g., Fast Point Feature Histogram (FPFH) [27], Fully Convolutional Geometric Features (FCGF) [11], D3Feat [10], Overlap Predator [28], GeoTransformer [29]). Since key-point matching results in high outlier rates, robust registration algorithms are required to estimate the registration parameters and obtain an approximate alignment (e.g., Random Sample Consensus (RANSAC) [9], TEASER++ [12], Spectral Matching [30], Fast Global Registration [31], Deep Global Registration [32], PointDSC [13]). Once an approximate alignment is obtained, local optimization is usually performed (e.g., [25, 33, 34]).

Robust registration algorithms are usually designed following the hypotheses generation and evaluation framework (e.g., [9, 13, 35, 36]): First, multiple hypotheses of the estimated transformation are generated, and at a second stage, this step is followed by evaluation of the hypotheses, selection, and refinement of the best fitting hypothesis.

The proposed registration backbone follows the hypothesis generation and hypothesis evaluation paradigm as well. However, unlike existing algorithms where hypotheses are generated from putative point correspondences and selected using the sample consensus criterion, in the proposed method hypotheses are generated using the RTUME

[5] from local *neighborhoods* of putative matches. Thus, hypotheses that are generated from disjoint neighborhoods provide independent estimates of the unknown, underlying transformation, a property that is used in the following hypothesis evaluation step, as detailed in Sect. 3.2. Hypotheses evaluation is performed using a unique multi-consensus analysis that provides high-confidence hypothesis selection, and a failure detection mechanism when no such consensus is found. Figure 5 illustrates the structure of the registration backbone.

3.1 The universal manifold embedding for hypotheses generation

The Rigid Transformation Universal Manifold Embedding (RTUME) [4, 37–39] is a methodology for constructing a matrix representation of an observation such that this representation is covariant with the transformation, and then using this representation to identify a linear subspace that is invariant to rigid coordinate transformations of the observation.

More specifically, consider a 3-D object s, and the *orbit* of equivalent objects formed by the action of the transformation group $G = SE(3)$. There exists one such orbit for each object s. Let $\mathcal{O}_s \subset \mathbb{R}^3$ be the set of all possible point cloud observations on elements in the orbit of s.

Let $o \in \mathcal{O}_s$ be a point cloud observation on some element from the orbit of s, and let $f : \mathbb{R}^3 \to \mathbb{R}^d$ be a function that assigns a real-valued vector to each point in the observation. We name f the *observation coloring function* and denote by $f_o(\mathbf{x}) = \{f(\mathbf{x})|\mathbf{x} \in o\}$ the *colored observation*. To simplify notations, let $h(\mathbf{x}) = f_o(\mathbf{x})$.

The *Rigid Transformation Universal Manifold Embedding* (RTUME) maps every observation h from the orbit of s to a matrix $\mathbf{T}(h) \in \mathcal{M}(M, 4)$, such that $\mathbf{T}(h)$ is **covariant** with the geometric rigid transformation, and where $\mathcal{M}(M, 4)$ is the space of $M \times 4$ real-valued matrices, and M the dimension of the embedding Euclidean space. The map

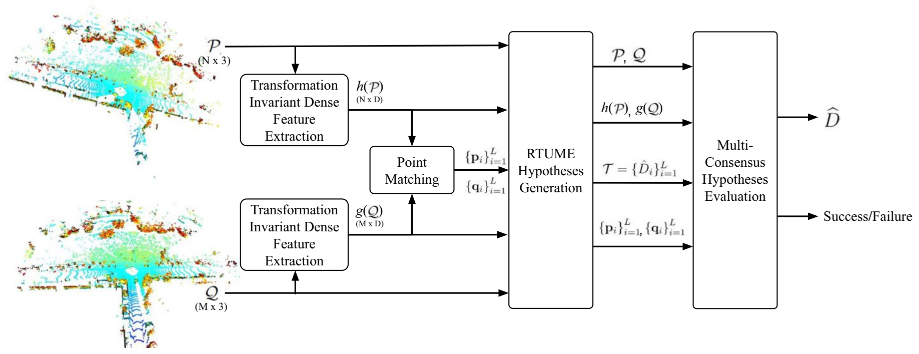

$\mathcal{P}$
(N x 3)
Transformation Invariant Dense Feature Extraction
$h(\mathcal{P})$
(N x D)
Point Matching
$\{\mathbf{p}_i\}_{i=1}^{L}$
$\{\mathbf{q}_i\}_{i=1}^{L}$
RTUME Hypotheses Generation
$\mathcal{P}, \mathcal{Q}$
$h(\mathcal{P}), g(\mathcal{Q})$
$\mathcal{T} = \{\hat{D}_i\}_{i=1}^{L}$
$\{\mathbf{p}_i\}_{i=1}^{L}, \{\mathbf{q}_i\}_{i=1}^{L}$
Multi-Consensus Hypotheses Evaluation
$\widehat{D}$
Success/Failure
Transformation Invariant Dense Feature Extraction
$g(\mathcal{Q})$
(M x D)
$\mathcal{Q}$
(M x 3)

Fig. 5 The proposed registration pipeline. Given two point clouds $\mathcal{P}$ and $\mathcal{Q}$, dense features are extracted on each point cloud, denoted by $h(\mathcal{P})$ and $g(\mathcal{Q})$. Assuming that the feature extraction function is SE(3)-invariant, the relation $h(\mathbf{p}) = g(D_0(\mathbf{q}))$ holds for any pair of corresponding points $\mathbf{p} \in \mathcal{P}$ and $\mathbf{q} \in \mathcal{Q}$, where $D_0 \in SE(3)$ is the underlying, unknown transformation between the point clouds. Then L putative point correspondences are obtained using ℓ_2 distances between the extracted features, denoted by $\{\mathbf{p}_i\}_{i=1}^{L}$ and $\{\mathbf{q}_i\}_{i=1}^{L}$. The point clouds, extracted features, and putative matches are used as input for the RTUME hypothesis generation module described in Sect. 3.1.3, resulting in a set of hypothesized transformation estimates $\mathcal{T} = \{\hat{D}_i\}_{i=1}^{L}$. Finally, $\mathcal{T}$, the extracted features, point clouds, and putative matches, are used in the multiple consensus hypotheses evaluation to select the best hypothesis, and an estimate of the success probability of the registration procedure, as described in Sect. 3.2

$\mathcal{G} : \mathcal{M}(M,4) \to \text{ Gr }(M,4)$, where $\text{Gr }(M,4)$ is the Grassmann manifold of 4-dimensional linear subspaces of M-dimensional Euclidean space, maps $\mathbf{T}(h)$ to its column space $\langle\mathbf{T}(h)\rangle$. Thus, the RTUME maps the orbit of s into a *single* G **-invariant** linear subspace $\langle\mathbf{T}(h)\rangle \in \text{ Gr }(M,4)$, as detailed below.

3.1.1 The RTUME descriptor

Given a real vector-valued 3-D observation h, the RTUME matrix representation of $h(\mathbf{x})$ is given by

$$\mathbf{T}(h) = \begin{bmatrix} \int\limits_{\mathbb{R}^3} w_1 \circ h(\mathbf{x})d\mathbf{x} & \int\limits_{\mathbb{R}^3} x_1 w_1 \circ h(\mathbf{x})d\mathbf{x} & \dots & \int\limits_{\mathbb{R}^3} x_3 w_1 \circ h(\mathbf{x})d\mathbf{x} \\ & \vdots & & \\ \int\limits_{\mathbb{R}^3} w_M \circ h(\mathbf{x})d\mathbf{x} & \int\limits_{\mathbb{R}^3} x_1 w_M \circ h(\mathbf{x})d\mathbf{x} & \dots & \int\limits_{\mathbb{R}^3} x_3 w_M \circ h(\mathbf{x})d\mathbf{x} \end{bmatrix} \tag{1}$$

where w_i, $i = 1, ..., M$ are measurable functions aimed at generating many compandings of the observation and x_k denotes the k-th element of $\mathbf{x}$. We note that since in the navigation application the observations are point clouds related by a rigid transformation, the integration measure in the elements of (1) is the counting measure. See [40]. A method for optimizing the design of w_i, $i = 1, ..., M$ is presented in Sect. 3.1.4.

Let $h(\mathbf{x})$ and $g(\mathbf{x})$ be two such "colored" point cloud observations related by a rigid transformation of coordinates such that $h(\mathbf{x}) = g(\mathbf{R}\mathbf{x} + \mathbf{t})$. The RTUME matrices $\mathbf{T}(h)$ and $\mathbf{T}(g)$ constructed from $h(\mathbf{x})$ and $g(\mathbf{x})$ as in (1) are related as follows

$$\mathbf{T}(h) = \mathbf{T}(g)\mathbf{D}^{-1}(\mathbf{R}, \mathbf{t}) \ , \tag{2}$$

where $\mathbf{D}(\mathbf{R}, \mathbf{t})$ is given by

$$\mathbf{D}(\mathbf{R}, \mathbf{t}) = \begin{bmatrix} 1 & \mathbf{t}^T \\ \mathbf{0} & \mathbf{R}^T \end{bmatrix} \tag{3}$$

We therefore conclude that for rigid transformations of 3-D objects, the RTUME is a mapping of functions (defined on the objects) to matrices, such that the mapping is covariant with the rigid coordinate transformation. In other words, the RTUME matrices of 3-D functions related by a rigid transformation are related by a re-expression of the same rigid transformation that relates the functions. A detailed derivation of the RTUME descriptor and estimator is given in [4].

3.1.2 SE(3)*-invariant point cloud coloring*

Point clouds are sets of coordinates in 3-D with no functional relation imposed on them. Hence, a necessary step in adapting the RTUME framework for point cloud processing is to define a function that assigns each point in the cloud with a value, invariant to the action of the transformation group. We call this step of generating SE(3)-invariant dense feature descriptor: SE(3)-*invariant coloring*. Moreover, since the point clouds to be registered may be acquired by different sensor modalities, at different times and from different points of view, the coloring function has to be robust to different sampling patterns, especially when registration of an observation to a previously generated reference map is considered. Since defining such a function analytically can be very hard, if at all possible, we adopt a data-driven approach for implementing the coloring function. This data-driven approach is based on the FCGF [11] architecture, where the goal is to encapsulate into the feature vector assigned to every point, global and local contextual information. In our implementation, the FCGF feature vector assigned to every point in the processed point cloud is a 32-dimensional vector. To improve the rotation invariance of the SE(3)-invariant coloring, the FCGF model is trained using rotational augmentations. For achieving better robustness to the differences in sampling rates between LiDAR observations and the reference map, both in the training stage and for the actual navigation, multiple consecutive scans are registered and aggregated into a larger point cloud fragment (8 in the experiments detailed in Sect. 4) to create observations with higher uniformity in their sampling.

3.1.3 Hypothesis generation using the RTUME descriptor

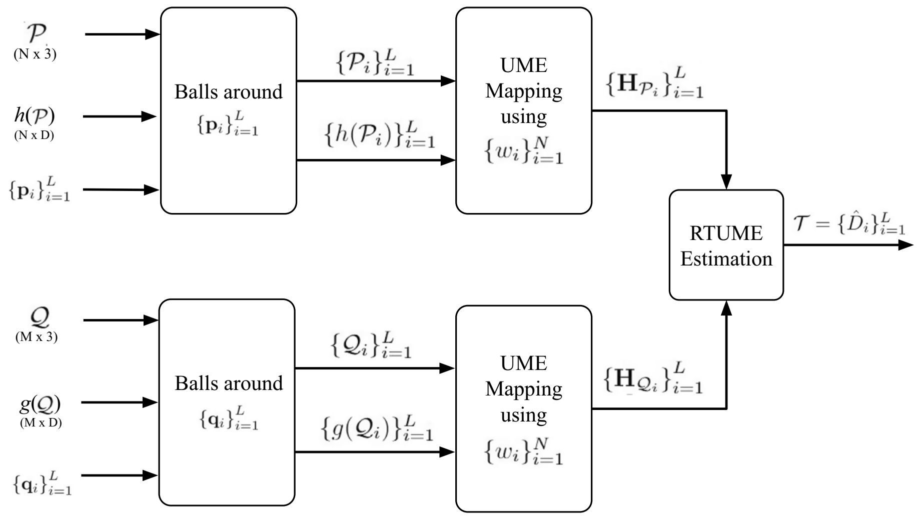
Let $\mathcal{Q}$ and $\mathcal{P}$ denote the observed and reference (target) point clouds, respectively. Let $h(\mathbf{x})$ denote a function defined on $\mathcal{P}$. In addition, define the neighborhoods of key points $\{\mathbf{p}_i\}_{i=1}^{L} \subset \mathcal{P}$, $\{\mathbf{q}_i\}_{i=1}^{L} \subset \mathcal{Q}$ in each point cloud as $\{\mathcal{P}_i\}_{i=1}^{L} \subset \mathcal{P}$, $\{\mathcal{Q}_i\}_{i=1}^{L} \subset \mathcal{Q}$ obtained by the intersection of radius R balls around each point. Given a point $\mathbf{p}_i \in \mathcal{P}$, the *local RTUME descriptor* of $\mathbf{p}_i$ is denoted by $\mathbf{H}_{\mathbf{p}_i}$. It is evaluated using a local adaptation of (1), such that the sums are evaluated locally on the subset $\mathcal{P}_k$. Thus, for correctly matched points, the relation (2) holds.

Given $\mathcal{P}$ and $\mathcal{Q}$, with putative matching key-point pairs $\{\mathbf{p}_i\}_{i=1}^{L}$ and $\{\mathbf{q}_i\}_{i=1}^{L}$ (where matches are determined based on ℓ_2 distances between the points feature vectors, or alternatively using the RTUME distances between their neighborhoods), we next evaluate the corresponding sets $\left\{\mathbf{H}_{\mathbf{p}_i}\right\}_{i=1}^{L}$ and $\left\{\mathbf{H}_{\mathbf{q}_i}\right\}_{i=1}^{L}$ of local RTUME descriptors using the corresponding local subsets $\{\mathcal{P}_i\}_{i=1}^{L}$ $\{\mathcal{Q}_i\}_{i=1}^{L}$.

For these sets of putative matching key points and their neighborhoods, denote the estimate of the local transformation between $\mathbf{H}_{\mathbf{p}_i}$ and $\mathbf{H}_{\mathbf{q}_i}$ evaluated using (2) by $\widehat{D}_i$ and the set of hypotheses as $\mathcal{T} = \{\hat{D}_i\}_{i=1}^{L}$.

By adapting the RTUME to a local key-point descriptor, key-point-to-key-point local transformation is simultaneously estimated with the distance between the key-point descriptors. For correctly matched key points, these local estimates are also estimates of the unknown underlying transformation between the point clouds. Therefore, the local transformation estimates corresponding to *correctly* matched key points form a cluster of rigid transformations among randomly distributed rigid transformations of false matches [5].

The partial overlap between the observed point cloud and the reference map segment, combined with the differences in their sampling patterns, poses a significant difficulty in the place recognition and localization initialization step, as high rates of false matches are encountered. Note that since these estimates are local, estimates obtained from disjoint neighborhoods are independent. Thus, this approach is robust in the presence of any number of outliers, as long as a minimal set of correct local estimates exists in $\mathcal{T}$. Figure 6 illustrates the hypothesis generation process.


$\mathcal{P}$ (N x 3)
$h(\mathcal{P})$ (N x D)
$\{\mathbf{p}_i\}_{i=1}^{L}$
Balls around $\{\mathbf{p}_i\}_{i=1}^{L}$
$\{\mathcal{P}_i\}_{i=1}^{L}$
$\{h(\mathcal{P}_i)\}_{i=1}^{L}$
UME Mapping using $\{w_i\}_{i=1}^{N}$
$\{\mathbf{H}_{\mathcal{P}_i}\}_{i=1}^{L}$
RTUME Estimation
$\mathcal{T}=\{\hat{D}_i\}_{i=1}^{L}$
$\mathcal{Q}$ (M x 3)
$g(\mathcal{Q})$ (M x D)
$\{\mathbf{q}_i\}_{i=1}^{L}$
Balls around $\{\mathbf{q}_i\}_{i=1}^{L}$
$\{\mathcal{Q}_i\}_{i=1}^{L}$
$\{g(\mathcal{Q}_i)\}_{i=1}^{L}$
UME Mapping using $\{w_i\}_{i=1}^{N}$
$\{\mathbf{H}_{\mathcal{Q}_i}\}_{i=1}^{L}$
Fig. 6 Hypothesis generation using the RTUME descriptor. First, dense features are extracted for each point cloud. These features are used both for obtaining putative point correspondence and as a "coloring" function for the RTUME. Then, RTUME matrices are constructed for each neighborhood, and hypothesized transformation estimates are generated using (2) for each putative match

3.1.4 Optimizing the design of The RTUME operator

A key issue in optimizing the design of the RTUME operator is the design of the set of nonlinear functions $\{w_i\}_{i=1}^{M}$ used in (1). While previous work dealt with the design of $\{w_i\}_{i=1}^{M}$ in the context of optimal classification performance [39, 41, 42], optimizing the design in the context of registration accuracy was an open problem. To preserve the invariance to the underlying transformation of coordinates, the nonlinear functions, $\{w_i\}_{i=1}^{M}$, are required to operate only on the "coloring" values assigned to the points of the point clouds. In case the coloring function assigns an M-dimensional feature *vector* to each point (as is the case when using FCGF), a straightforward solution for designing the nonlinear functions $\{w_i\}_{i=1}^{M}$ is to set $w_i \circ h(\mathbf{x})$ to be the ith element of the feature vector $h(\mathbf{x})$. However, to provide an optimized learned solution to this design problem, a multilayer perceptron (MLP) is designed for implementing these continuous nonlinear functions. The MLP input is the set of dense features assigned to each point (i.e., $h(\mathbf{x}), \mathbf{x} \in \mathcal{P}$). Its output is a corresponding vector whose elements are $w_i \circ h(\mathbf{x})$ for every point in the point cloud. Thus, the ith element in the output vector of the MLP is the result of applying the ith nonlinear function w_i to $h(\mathbf{x})$.

To train this architecture, the empirical mean and standard deviation of local estimates $\widehat{D}_i$ is evaluated using a distance measure defined on SE(3) [43], as follows. SE(3) is the Cartesian product $SO(3) \times \mathbb{R}^3$. Therefore, distances between elements of SE(3) are usually measured by heuristic combinations of the rotation distance and translation distance. However, these distances have different scales which makes these approaches improper for rigorous analysis of the distances between SE(3) elements. Therefore, in the following we adopt a metric that naturally combines the angular and translation distances to measure distances between SE(3) elements: Let $C \subset \mathbb{R}^3$ be a set of predetermined points, used to measure the distance between transformations (e.g., vertices of a (possibly scaled) unit cube). Given two rigid transformations, $D_1, D_2 \in SE(3)$, we define the distance between them with respect to C as

$$d_C(D_1, D_2) = \sqrt{\frac{1}{|C|}\sum_{c\in C}\|D_1(c) - D_2(c)\|^2} \tag{4}$$

Having a distance metric, a common practice for evaluating the sample average of group elements is to define the sample average as the group element which minimizes the distance to all other elements in the sample. Let $\mathcal{T} = \{\widehat{D}_i\}_{i=1}^{L}$ be a set of rigid transformations. The mean transformation of $\mathcal{T}$ with respect to the distance (4) is obtained by solving

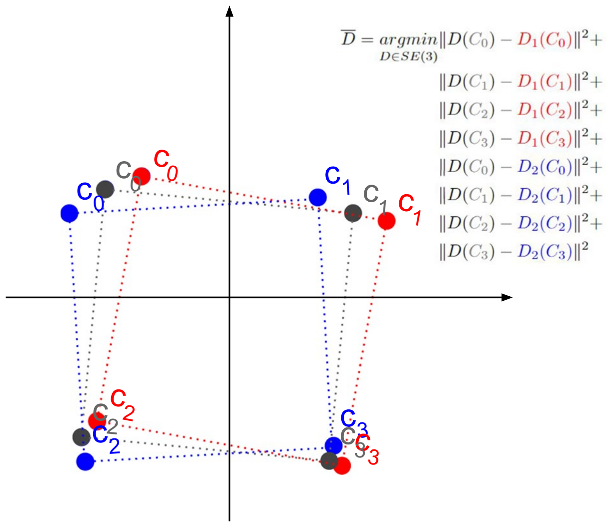
$$\bar{D} = \underset{D\in \mathrm{SE}(3)}{\arg\min} \sum_{i=1}^{N} d_C(D, \hat{D}_i) \tag{5}$$

Since the sets of transformations and points are known, the right-hand side of (5) is equivalent to the least squares problem solved in [44] while using "duplicates" of C for each element in $\mathcal{T}$. Thus, a closed-form solution to the problem of estimating the mean transformation is obtained. Figure 7 depicts the defined least squares problem, solved in (5) in 2D.

Once the empirical mean is found, the sum of squared distances of each local estimate from the average local estimate is an estimate of the error variance.

$$\sigma_{\mathcal{T}}^2 = \frac{1}{|\mathcal{T}|}\sum_{i=1}^{N} d_C^2(\overline{D}, \widehat{D}_i) \tag{6}$$

The average transformation error and estimated variance are used in a loss function to optimize a set of nonlinear functions w_i that define the RTUME operator (1), by



$\overline{D} = \underset{D\in SE(3)}{argmin}\|D(C_0) - D_1(C_0)\|^2 +$
$\|D(C_1) - D_1(C_1)\|^2 +$
$\|D(C_2) - D_1(C_2)\|^2 +$
$\|D(C_3) - D_1(C_3)\|^2 +$
$\|D(C_0) - D_2(C_0)\|^2 +$
$\|D(C_1) - D_2(C_1)\|^2 +$
$\|D(C_2) - D_2(C_2)\|^2 +$
$\|D(C_3) - D_2(C_3)\|^2$
C_0
C_1
C_2
C_3

Fig. 7 Illustration of the least squares problem, solved in (5). Blue and red points denote the results of applying two sample elements of $\mathcal{T}$ on the points of C. Gray points denote the result of applying the (searched) target transformation on the points in C. The transformation that minimizes the sum of squared distances is the average transformation

employing metric learning. Using transformation estimates from inlier matches, the average estimation error and estimated variance are minimized (positive loss), while the empirical variance of transformation estimates associated with outlier matches is maximized (negative loss). More specifically, let D_0 be the unknown underlying transformation, $\mathcal{T}_{\text{pos}}$ the set of estimates associated with inlier matches with mean and variance $\overline{D}_{\text{pos}}, \sigma^2_{\mathcal{T}_{\text{pos}}}$, respectively, and $\mathcal{T}_{\text{neg}}$ the set of estimates associated with outlier matches, with variance $\sigma^2_{\mathcal{T}_{\text{neg}}}$. The positive loss function is given by

$$\mathcal{L}_{\text{pos}} = d_C(\overline{D}_{\text{pos}}, D_0) + \sigma_{\mathcal{T}_{\text{pos}}}, \tag{7}$$

while the negative loss only considers the standard deviation of transformations estimated from outlier matches,

$$\mathcal{L}_{\text{neg}} = \sigma_{\mathcal{T}_{\text{neg}}} \tag{8}$$

The purpose of the negative loss is to minimize the probability of finding an SE(3) consensus set (see Sect. 3.2.2) originating from outlier matches, by driving incorrect estimates away from each other. Thus, the complete loss function to be minimized is $\mathcal{L} = \mathcal{L}_{\text{pos}} - \alpha \mathcal{L}_{\text{neg}}$, with α being a small weight constant.

3.2 Hypothesis evaluation and selection using multiple consensus analysis

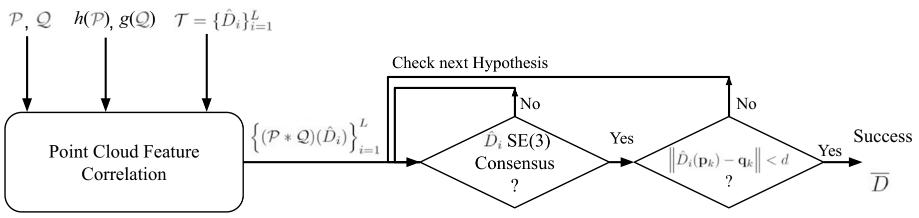
In this section, we present a robust multi-consensus hypothesis evaluation method, incorporating the point cloud features correlation (PCFC), an SE(3) consensus from independent local estimates, and a key-point consistency test. This approach dramatically decreases the probability of choosing a false estimate while providing a failure detection method indicating when none of the estimated hypotheses confers with the set of predetermined constraints.

According to the proposed multiple consensus test, a hypothesis $\widehat{D}_i$ is required to maximize the PCFC (or be one of the hypotheses that maximize the PCFC), detailed in Sect. 3.2.1. Thus, the first step is to evaluate the PCFC for each of the generated hypotheses. Then, ordered by descending PCFC values, we check if the evaluated hypothesis has a sufficiently large consensus size in the SE(3) consensus of independent estimates of Sect. 3.2.2. Then, a final check is performed by verifying that the matching key points that resulted in the SE(3) consensus are indeed transformed pointwise, by $\widehat{D}_i$ from the observation to the reference map (up to a small predetermined threshold). If no hypothesis passes the three-stage test, the registration is determined to be a failure.

Figure 8 illustrates the components and flow of the multiple consensus criterion. Figure 9 illustrates an example of a registration problem from the 3DMatch data set [23], showing a multiple consensus set of key points with the local neighborhoods from which RTUME matrices were obtained.

3.2.1 Point clouds feature correlation

In the RTUME hypothesis generation setup (see Sect. 3.1.3), a set of hypotheses is generated using the RTUME matrix representation (1), from local neighborhoods of putative matches. These transformation hypotheses contain both correct (up to a threshold) and incorrect estimates. An essential step in the registration process is to sort the correct



$\mathcal{P}$, $\mathcal{Q}$
$h(\mathcal{P})$, $g(\mathcal{Q})$
$\mathcal{T} = \{\hat{D}_i\}_{i=1}^{L}$
Point Cloud Feature Correlation
$\left\{(\mathcal{P} * \mathcal{Q})(\hat{D}_i)\right\}_{i=1}^{L}$
Check next Hypothesis
No
$\hat{D}_i$ SE(3) Consensus ?
Yes
No
$\left\|\hat{D}_i(\mathbf{p}_k) - \mathbf{q}_k\right\| < d$?
Yes
Success
$\overline{D}$

Fig. 8 The multiple consensus evaluation algorithm

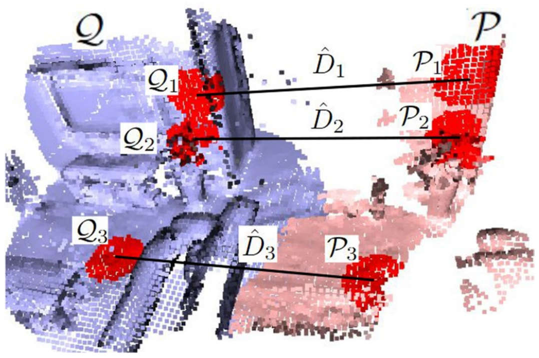


Fig. 9 Multiple consensus principle: Example taken from a registration problem of the 3DMatch data set. Three independent estimates are shown. They result from disjoint neighborhoods of matching key points (colored in red)

hypotheses out of the generated set of hypotheses. This is commonly performed using the sample consensus criterion, introduced in [9], and is being employed by many state-of-the-art robust registration methods (e.g., [13, 35]). A major drawback of any sample consensus approach is that regardless of the algorithm's ability to generate accurate hypotheses, if a false hypothesis achieves higher sample consensus than the correct hypothesis, the latter cannot be detected. Thus in the proposed registration pipeline, given a set of noisy transformation estimates $\{\hat{D}_i\}_{i=1}^{L} \in SE(3)$ instead of making a decision based exclusively on Euclidean distances between the assumed matching points, a correlation criterion that correlates the geometric and contextual properties of the entire point clouds is employed [6], as it allows to choose the correct transformation even when the hypothesis that results in the largest sample consensus is false. In the presence of false consensus sets this approach provides better robustness and accuracy than the standard consensus size criterion. Next, this criterion is briefly reviewed.

Recall that following the dense feature extraction, every point of the point cloud is assigned with an M-dimensional real-valued feature vector, encapsulating into the vector global and local contextual information. Define the point cloud feature correlation (PCFC) between two point clouds $\mathcal{P} \subset \mathbb{R}^3$ and $\mathcal{Q} \subset \mathbb{R}^3$, as function of an arbitrary transformation $D \in SE(3)$, using feature functions $h(\mathbf{x})$ and $g(\mathbf{x})$ defined on $\mathcal{P}$ and $\mathcal{Q}$, respectively, as follows:

$$(\mathcal{P} * \mathcal{Q})(D) = \sum_{\mathbf{p} \in \mathcal{P}} k(h(\mathbf{p}), g(D^{-1}(\mathcal{Q}))) \tag{9}$$

where k is a weighted correlation function of a single point, defined by

$$k(h(\mathbf{p}), g(\mathcal{Q})) = \sum_{\mathbf{q}\in\mathcal{Q}} w_\sigma(\|\mathbf{p}-\mathbf{q}\|)h(\mathbf{p})^T g(\mathbf{q}) \tag{10}$$

$w_\sigma : \mathbb{R}^+ \to [0,1]$ is a weight function, inversely related to the distance between given points. In [6], (9) is used to measure the quality of a transformation estimate, where the goal is to find the transformation $\widehat{D}_0$ that maximizes (9), instead of the commonly employed consensus size criterion. Let $\mathcal{T} = \{\hat{D}_i\}_{i=1}^{K}$ be a set of hypothesis estimates. The estimate of the underlying transformation D_0 using (9) is given by

$$\hat{D}_0 = \arg\max_{D\in\mathcal{T}}(\mathcal{P} * \mathcal{Q})(D) \tag{11}$$

However, in this work instead of simply choosing the best hypothesis using (11), (9) is used to rank the best hypotheses in terms of PCFC, to initiate a search for an SE(3) consensus set of independently generated hypotheses.

3.2.2 SE(3) consensus of independent estimates

In Sect. 3.1.3, the RTUME for estimating transformations between local neighborhoods of putative matches was presented. When these neighborhoods are disjoint, the local estimates are independent (see Fig. 9). This property is employed in an additional step aimed at mitigating the devastating effect of outlier estimates, by using the SE(3) consensus of independent estimates, defined next: A candidate hypothesis $\hat{D}_i$ is accepted if there exist at least n independent hypotheses (i.e., that originated from disjoint neighborhoods) such that $d_C(\hat{D}_k, \hat{D}_i) < \alpha$, $k = 1, ..., n$, where α is a distance threshold determined by the required accuracy of local hypotheses. Next, once an SE(3) consensus of independent estimates is found, all estimates in the SE(3) neighborhood of $\hat{D}_i$ are averaged using (5) to provide an estimate of the underlying transformation.

4 Experimental results

4.1 Loop closure experiments

The problem of loop closure is defined as identifying a return of the robot to a previously visited place in the same drive. Loop closure is used in SLAM algorithms to correct the accumulated drift retrospectively. In terms of the problem definition, however, place recognition and loop closure solve the same kind of problem—identifying the location of a scan with respect to a dictionary of scans or submaps. Thus, we demonstrate the operation of the proposed place recognition module on the KITTI loop closure benchmark. More specifically, the vehicle in sequence 0 of the KITTI odometry data set [8] drives through parts of the route multiple times (see Fig. 10).

In the test, scans taken from locations where the vehicle visited more than once are used as query scans. Past scans of the query are used as the reference dictionary. State-of-the-art detection recall on this dataset is approximately 90% [15], where the proposed algorithm achieves 100% detection recall when using the optimized RTUME. In the implementation of the proposed PCMN we employ [15] to sort the reference dictionary for a quicker search. Since in the proposed algorithm registration is performed jointly with detection, only partial overlap between reference scans and the query scan

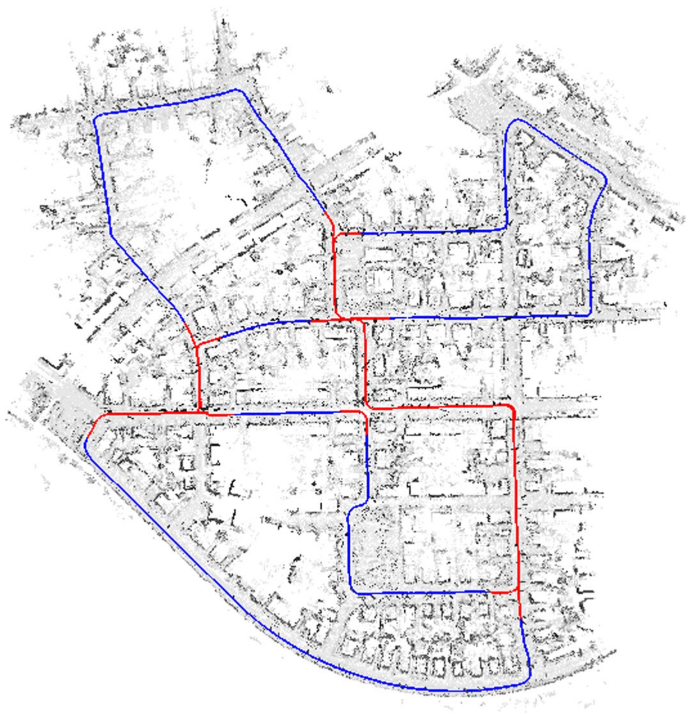


Fig. 10 Vehicle trajectory on sequence 00 from the KITTI odometry data set. Marked in red are locations where the vehicle passed multiple times

Table 2 First choice average recall (AR) and registration accuracy on the KITTI loop closure dataset, using various thresholds for localization success

Method	AR@25m	AR@0.5m	All		Successful	
			RE (deg)	TE (m)	RE (deg)	TE(m)
MinkLoc3DV2 + RANSAC	0.87	0.84	14.6	24.2	1.2	0.3
MinkLoc3DV2 + PointDSC	0.88	0.88	9.2	24.0	0.2	0.05
OverlapTrans. + RNASAC	0.90	0.85	12.5	27.4	1.8	0.4
OverlapTrans. + PointDSC	0.90	0.90	9.6	26.8	0.2	0.06
ScanContext + RANSAC	0.85	0.83	15.5	28.3	1.4	0.3
ScanContext + PointDSC	0.85	0.85	9.1	27.9	0.2	0.05
PCMN	**1.0**	**1.0**	**0.3**	**0.06**	0.3	0.06

The "Successful" column provides registration accuracy statistics on those cases where Place Recognition is considered successful. "All" column provides the overall registration accuracy statistics of the test

Bold highlights the method that achieves the best performance

is required to obtain accurate localization. Thus, we use only a fraction of the reference scans (using only 178 reference segments out of 4541). Table 2 provides loop closure performance and registration accuracy comparison between the precision localization module of PCMN and state-of-the-art place recognition methods: OverlapTransformer, [15], MincLoc3DV2 [22], and Scan Context [14]. Since navigation applications require precision localization, each of the tested place recognition algorithms is augmented by a registration step to refine the place recognition estimate. For the registration step, RANSAC [9] and PointDSC [13] were used to estimate the transformation based on the same set of putative correspondences employed by PCNM. Registration accuracy is reported over the complete dataset, as well as over only the successful attempts (i.e., where the place recognition error is less than 25 m). In addition, only the first choice (i.e., @1) average recall is presented, but using different thresholds to what is considered as "success": 25m for the place recognition, and 0.5 m for the refined registration estimate. It is concluded from Table 2 that, indeed, the limiting factor of precision localization is the likelihood of successful place recognition, as the registration accuracy is similar over all the compared methods when considering only the successful place recognition attempts. Current state-of-the-art OverlapTransformer place recognition achieves on this data sets only 0.9 recall at 25m. When paired with a modern registration algorithm, the refined average recall remains at 0.9 at 0.5m as well (i.e., precision place recognition fails in 10% of the attempts even after employing the registration step), while when using the classic RANSAC, average recall drops to 0.85 at 0.5m. On the other hand, the proposed PCMN achieves 1.0 recall, with the best overall registration accuracy – significantly better than state of the art. Thus, it is concluded that the proposed PCMN is better suitable for navigation applications as it is significantly more robust than the alternatives while providing the high registration accuracy required in these applications. Figure 11 depicts visual examples where the proposed PCMN was successful and the other tested methods failed.

4.2 Place recognition on KITTI

Unlike the loop closure task, where each scan is compared to a previously acquired set of scans to detect loops, in the place recognition task, a scan is compared to a complete reference map. Using the KITTI odometry data set, we generate a reference map by merging scans using the ground truth transformation, while removing all dynamic objects from the map (vehicles and people) using the ground truth semantic labels from the Semantic KITTI dataset [45]. Then, the reference map is partitioned into partially

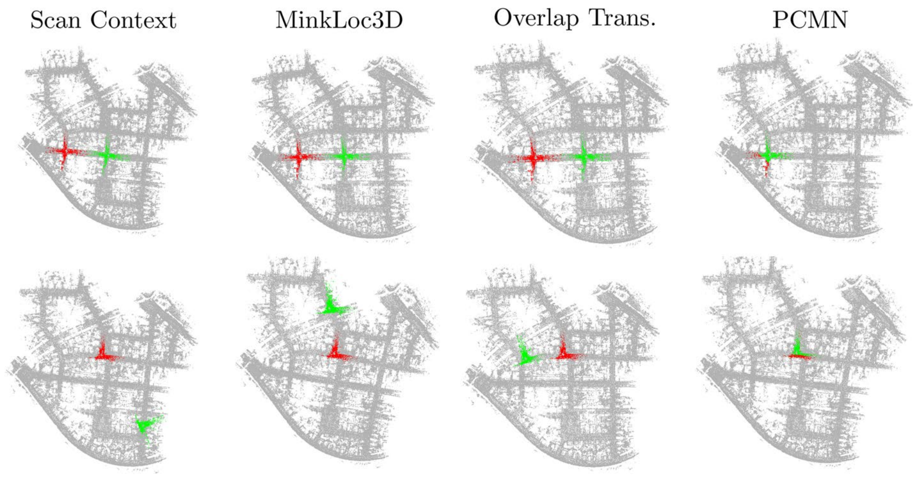


Fig. 11 Precision localization examples, depicting the first choice registration results using Scan Context, MinkLoc3D, Overlap Transformer, followed by pose refinement using PointDSC, and the proposed PCMN. The correct location of the query scan is colored in red, while the estimated location is colored in green. The localization results are shown over the complete reference (colored in gray). In the presented examples, only PCMN was successful in performing accurate registration

overlapping segments. The task is then to localize a scan (or a sequence of scans) within the reference map to initialize tracking with respect to the reference map. For better robustness to the differences between the sampling patterns of the scans and the reference map segments, multiple consecutive scans are registered and aggregated into a larger point cloud fragment (8 in the current implementation). In the test, for each query, the reference segments are sorted using Scan Context [14], as it performs best at this setup. Table 3 summarizes the results of the place recognition test on the KITTI data set. The test was performed on sequences 8, 9, and 10. The reference map of the area of each sequence contains 140, 80, and 50 segments, respectively. For each sequence, 100 queries were attempted. Place recognition was successful in approximately 97% of the attempts, with submesh resolution average (30 cm) translation error, and approximately 2^o average rotation error. ScanContext, which is employed by PCMN to sort the reference segments in order to speed-up the search, while achieves better performance than the other compared methods, performs poorly when evaluated on the full place recognition task with approximately 50% registration recall.

4.3 Outdoors navigation example

For the outdoor navigation example, we use the same reference map from the place recognition dataset described in Sect. 4.2. However, instead of performing only place recognition attempts, a query is first localized using the place recognition scheme, followed by sequential tracking with respect to the reference map, of the raw Velodyne data, using the 3DMMD [26]. Thus, this experiment demonstrates the full PCMN operation, from initialization to drift-free tracking. The initialization and tracking experiment was repeated using random initial locations, followed by tracking along 100 meters in each experiment. Table 4 summarizes the initialization and tracking performance. For comparison purposes, 3DMMD was replaced by ICP for tracking each frame with respect to the reference map. ICP and 3DMMD performed similarly in terms of accuracy. However, ICP failed occasionally, losing the location and accumulating large errors. When differential tracking (odometry, using no reference map) was attempted, it resulted with larger errors due to the accumulated drift. Figure 12 illustrates a drift-free map-based tracking using the 3DMMD, and the drift accumulated using differential tracking.

Table 3 Place recognition on the KITTI data set

Method	AR@1m	All		Successful	
		RE(deg)	TE(m)	RE(deg)	TE(m)
MinkLoc3DV2 + RANSAC	0.05	104.6	243.1	2.1	0.54
MinkLoc3DV2 + PointDSC	0.26	67.9	238.3	1.4	0.21
OverlapTrans. + RANSAC	0.02	111.5	243.8	2.4	0.54
OverlapTrans. + PointDSC	0.22	67.7	235.5	1.6	0.21
ScanContext + RANSAC	0.1	84.5	135.5	2.6	0.54
ScanContext + PointDSC	0.56	37.9	128.2	1.2	0.16
PCMN	**0.97**	**3.1**	**11.4**	1.5	0.25

The "Successful" column provides registration accuracy statistics on those cases where place recognition is considered successful. "All" column provides the overall registration accuracy statistics of the test

Bold highlights the method that achieves the best performance

Table 4 Average and 99 percentile errors using the proposed Point Cloud Map-Based Navigation (PCMN) on the KITTI data set

Method	**Avg.**		**99 prc.**	
	RE (deg)	**TE (m)**	**RE (deg)**	**TE (m)**
Initialization	1.0	0.15	2.1	0.21
Tracking (3DMMD)	0.9	0.09	2.3	0.20
Tracking (ICP)	0.9	1.54	3.9	5.3
Tracking (differential odometry)	1.7	1.34	6.8	7.4

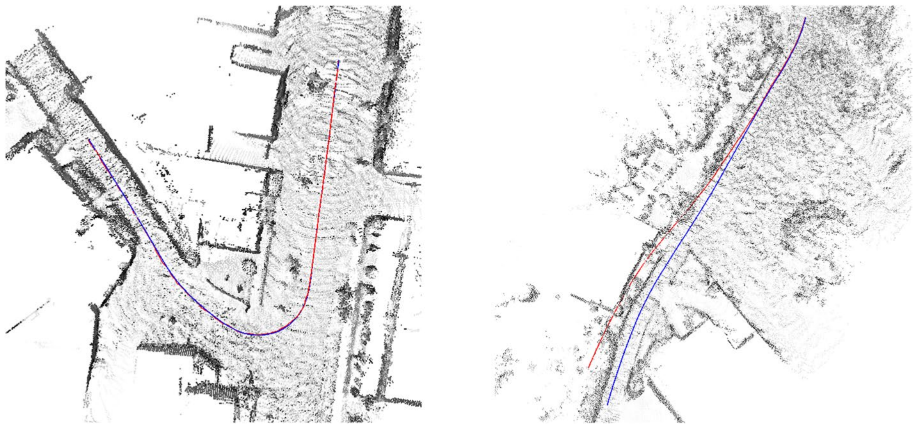


Fig. 12 Outdoor navigation example using the proposed PCMN on the KITTI dataset. Ground truth trajectory is shown in blue, and estimated trajectory is shown in red. Initialization was identical in all examples (see Sect. 4.2). Left: Drift-free map-based tracking using the 3DMMD. Right: Accumulated drift using differential tracking

4.4 Indoor place recognition

In this experiment, place recognition performance is evaluated, i.e., when the location is completely unknown. In the place recognition experiment, fragments are located within an indoor reference point cloud of the entire scene (see Fig. 13). To the best of our knowledge, there are no indoor place recognition algorithms that operate on point clouds. Thus, standard registration algorithms are employed as a baseline for evaluating the place recognition phase.

The reference map and query fragments were created from the 3DMatch [23] raw data, as follows: The raw scans were split into two disjoint sets for the generation of the reference map and for generating individual fragment observations. The reference map was created with TSDF implementation in Open3D API [46] using the ground truth transformations, covering the complete scene. Query fragments for the place recognition task were generated by merging consecutive scans with TSDF, covering a camera movement of approximately 25cm. Both the reference map and query fragments were created with a mesh resolution of 5cm for each of the 8 test scenes of the 3DMatch data set. In the implementation of all the compared methods, features are extracted using a publically available pre-trained FCGF model [11], where point correspondences were estimated using nearest neighbor search on L_2 distances between features.

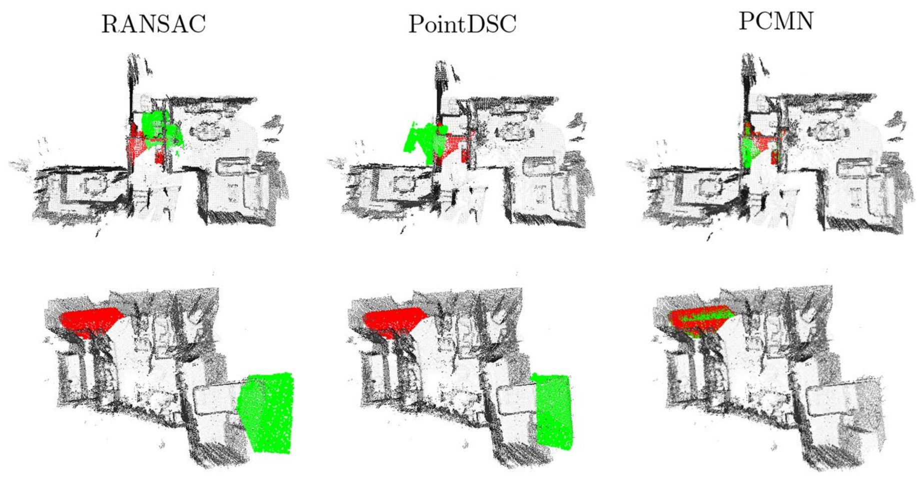


Fig. 13 Precision place recognition on the 3DMatch dataset, using standard registration algorithms and the proposed PCMN. The reference map is colored in gray, the ground truth location of the scan is marked in red, and the estimated registration result in green. In the examples, PCMN was successful in locating the precise location, while the compared methods failed

Table 5 Place recognition on the 3DMatch data set

Method	AR@0.3m	All		Successful	
		RE(deg)	TE(m)	RE(deg)	TE(m)
PointDSC	0.91	12.2	0.31	1.8	0.06
RANSAC	0.33	51.8	1.43	5.4	0.14
PCMN	0.95	9.0	0.13	0.7	0.03

The "Successful" column provides registration accuracy statistics on those cases where place recognition is considered successful. "All" column provides the overall registration accuracy statistics of the test

Table 5 details the precision place recognition experiment results. It can be seen that the proposed PCMN achieves the best first candidate average recall at a threshold of 30cm localization error. We also note that Success/Failure detection precision was 0.997 in this experiment, i.e., in almost every case where the Success/Failure decision module made a decision that the registration was successful—it was correct. This is in contrast to using the alternative methods that fail unexpectedly in approximately 10% of the cases for the modern PointDSC, and in 67% of the cases when using RANSAC. Figure 13 depicts examples of precision place recognition using the compared methods.

4.5 Indoor navigation example

For the indoor navigation example, the complete PCMN navigation pipeline is implemented for the 3DMatch data set. Tracking is performed with respect to the reference map, initialized by the proposed precision place recognition, starting at random locations within the reference map. After initialization, tracking is performed by performing local registration (ICP or 3DMMD) between the raw stream of scans and the reference map, initialized at the estimated location. Differential tracking (odometry) using ICP is presented for reference, as well.

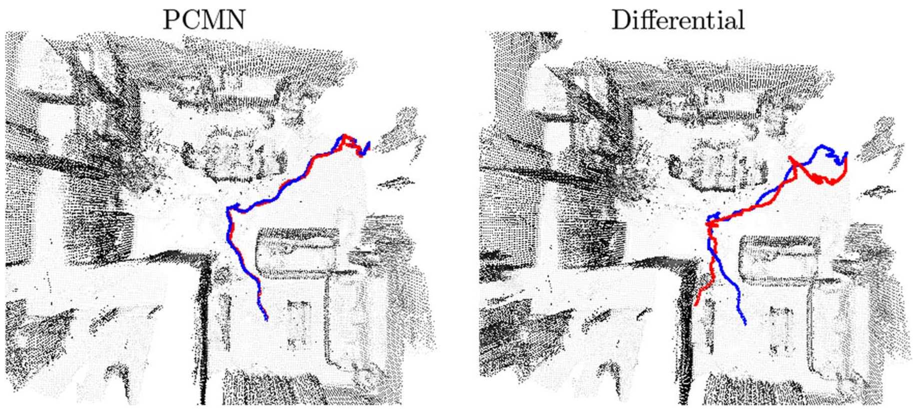


Fig. 14 Ground truth and estimated trajectories along the robot path using the proposed PCMN (left) versus differential tracking (right). The ground truth trajectory is marked in blue, while the estimated trajectory is marked in red

Table 6 Average and 99 percentile errors using the proposed Point Cloud Map-Based Navigation (PCMN) on the 3DMatch data set

Method	Avg.		99 prc.	
	RE (deg)	TE (m)	RE (deg)	TE (m)
Initialization (Place Recognition)	2.0	0.07	8.3	0.33
Tracking (3DMMD)	1.3	0.06	5.2	0.26
Tracking (ICP)	1.2	0.06	10.2	0.79
Tracking (differential odometry)	17.8	0.44	104.8	2.56

The reference map and query fragments were created as in Sect. 4.4, where the raw stream of scans was used as the input of the navigation system.

Table 6 details the navigation experiment results. It is concluded from the table that indeed the drift-free tracking accuracy using the PCMN is superior to differential tracking, with the 3DMMD achieving slightly favorable results over ICP with lower maximal error (similarly to the KITTI navigation example in Sect. 4.3). Figure 14 depicts an example of drift-free tracking versus differential tracking after the initialization phase.

4.6 Real time considerations

The proposed method is composed of precision place recognition and tracking modules. The tracking module is implemented by a local, and hence computationally efficient, registration algorithm, where instead of differential tracking between consecutive frames, registration is performed with respect to a reference point cloud. While in the performed experiments, ICP and 3DMMD were tested, there are alternative implementations of similar algorithms suitable for real-time applications. The precision place recognition implementation includes a coarse place recognition algorithm to sort the candidate map segments of the reference map, followed by registration attempts for eliminating non-matching segments and for accurately estimating the correct location with respect to the matching segment. Thus, the precision place recognition running time depends both on the coarse place recognition running time and accuracy. (The more accurate the coarse place recognition, the fewer registration attempts are required.) We note that since the registration attempt of the observation to every candidate map segment is independent of the registration attempts to the alternative candidates, this step is highly parallelizable.

In the performed experiments, all algorithms were implemented in Python (and thus were not optimized for running time). The precision localization average running time was in the order of 10 seconds. Since the precision place recognition validation step is executed in the background (see Fig. 2), it is not a limiting factor in real-time applications.

5 Conclusions

We have presented Point Cloud Map-based Navigation (PCMN), a drift-free autonomous navigation solution based exclusively on a novel robust point cloud registration backbone between 3-D scans and a known 3-D reference map. The registration backbone is equipped with a successful/failed registration detection. Registration is performed by generating multiple independent hypotheses from local neighborhoods of putative matching key points, using the rigid transformation universal manifold embedding. Then, hypotheses are evaluated and selected using a novel multiple consensus analysis, requiring the selected hypothesis to maximize the point clouds feature correlation as well as to be in consensus with other independent estimates.

When the robot location is unknown, detection of successful/failed registration is used in the initialization step (i.e., place recognition step) to detect overlap between a query scan and a reference map segment, by assuming that a "successful" registration is obtained only between point clouds with sufficient overlap. After initialization, tracking is performed directly with respect to the already known reference map segment, by *locally* maximizing the point cloud feature correlation on the SE(3) manifold. Since tracking is performed directly with respect to the 3-D reference map, no drift is accumulated during tracking.

The proposed PCMN was demonstrated on multiple tasks, including loop closure detection, place recognition with respect to a reference map, and indoor and outdoor navigation, achieving high localization recall and accuracy.

Acknowledgements
Not applicable.

Author contributions
This manuscript is part of AE PhD research under the supervision of Prof. JMF.

Funding
This research is supported by Israel Innovation Authority Grant 77887. The funding body had no role in the design, analysis, and interpretation of this work.

Availability of data and materials
The data used in this research are from publicly available sources.

Declarations

Competing interests
There are no conflict of interest.

Received: 31 October 2023 Accepted: 12 April 2024
Published online: 30 April 2024

References

1. J. Komorowski, Minkloc3d: Point cloud based large-scale place recognition. in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1790–1799 (2021)

2. Y. Xia, Y. Xu, S. Li, R. Wang, J. Du, D. Cremers, U. Stilla, Soe-net: a self-attention and orientation encoding network for point cloud based place recognition. CVPR **2021**, 11348–11357 (2021)
3. D. Rozenberszki, A. Majdik, LOL: Lidar-only Odometry and Localization in 3D point cloud maps. in *2020 IEEE International Conference on Robotics and Automation* (ICRA) (2020). IEEE
4. A. Efraim, J.M. Francos, Estimating rigid transformations of noisy point clouds using the universal manifold embedding. J. Math. Imaging Vision **64**(4), 343–363 (2022). https://doi.org/10.1007/s10851-022-01070-6
5. A. Efraim, J.M. Francos, Dual transformation and manifold distances voting for outlier rejection in point cloud registration. in *Proceedings of the IEEE Conference on Computer Vision*, pp. 4204–4212 (2021)
6. A. Efraim, J.M. Francos, On minimizing the probability of large errors in robust point cloud registration. IEEE Open J. Signal Process. **5**, 39–47 (2024). https://doi.org/10.1109/OJSP.2023.3340111
7. A. Efraim, J.M. Francos, The universal manifold embedding for estimating rigid transformations of point clouds. in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5157–5161 (2019). IEEE
8. A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the KITTI vision benchmark suite. in *Conference on Computer Vision and Pattern Recognition (CVPR)* (2012)
9. M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981). https://doi.org/10.1145/358669.358692
10. X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, C.-L. Tai, D3feat: Joint learning of dense detection and description of 3d local features. in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6358–6366 (2020). https://doi.org/10.1109/CVPR42600.2020.00639
11. C. Choy, J. Park, V. Koltun, Fully convolutional geometric features. in *ICCV* (2019)
12. H. Yang, J. Shi, L. Carlone, Teaser: fast and certifiable point cloud registration. IEEE Trans. Rob. **37**(2), 314–333 (2021). https://doi.org/10.1109/TRO.2020.3033695
13. X. Bai, Z. Luo, L. Zhou, H. Chen, L. Li, Z. Hu, H. Fu, C.-L. Tai, PointDSC: Robust point cloud registration using deep spatial consistency. in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15854–15864 (2021). https://doi.org/10.1109/CVPR46437.2021.01560
14. G. Kim, A. Kim, Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map. in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4802–4809 (2018). https://doi.org/10.1109/IROS.2018.8593953
15. J. Ma, J. Zhang, J. Xu, R. Ai, W. Gu, X. Chen, Overlaptransformer: an efficient and yaw-angle-invariant transformer network for lidar-based place recognition. IEEE Robot. Autom. Lett. **7**(3), 6958–6965 (2022). https://doi.org/10.1109/LRA.2022.3178797
16. M.A. Uy, G.H. Lee, Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018)
17. C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660 (2017)
18. R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, J. Sivic, Netvlad: Cnn architecture for weakly supervised place recognition. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)
19. W. Maddern, G. Pascoe, C. Linegar, P. Newman, 1 year, 1000 km: the oxford robotcar dataset. Int. J. Robot. Res. **36**(1), 3–15 (2017)
20. W. Zhang, C. Xiao, Pcan: 3d attention map learning using contextual information for point cloud based retrieval. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)
21. Z. Liu, S. Zhou, C. Suo, P. Yin, W. Chen, H. Wang, H. Li, Y.-H. Liu, Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis. in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019)
22. J. Komorowski, Improving point cloud based place recognition with ranking-based loss and large batch training. in *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 3699–3705 (2022). https://doi.org/10.1109/ICPR56361.2022.9956458
23. A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, T. Funkhouser, 3dmatch: Learning local geometric descriptors from rgb-d reconstructions (2017)
24. D. Chetverikov, D. Svirko, D. Stepanov, P. Krsek, The trimmed iterative closest point algorithm. Int. Conf. Pattern Recognit. **16**, 545–5483 (2002). https://doi.org/10.1109/ICPR.2002.1047997
25. P.J. Besl, N.D. McKay, A method for registration of 3-d shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992). https://doi.org/10.1109/34.121791
26. A. Efraim, J.M. Francos, 3D matched manifold detection for optimizing point cloud registration. in *ICECCME* (2022). https://doi.org/10.1109/ICECCME55909.2022.9988221
27. R.B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (fpfh) for 3d registration. in *2009 IEEE International Conference on Robotics and Automation*, pp. 3212–3217 (2009). https://doi.org/10.1109/ROBOT.2009.5152473
28. S. Huang, Z. Gojcic, M. Usvyatsov, K.S. A. Wieser, PREDATOR: Registration of 3d point clouds with low overlap. in *CPVR* (2021)
29. Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, K. Xu, Geometric transformer for fast and robust point cloud registration. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11143–11152 (2022)
30. M. Leordeanu, M. Hebert, A spectral technique for correspondence problems using pairwise constraints. in *ICCV*, pp. 1482–14892 (2005). https://doi.org/10.1109/ICCV.2005.20
31. Q.Y. Zhou, J. Park, V. Koltun, Fast global registration. in *ECCV* (2016)
32. C. Choy, W. Dong, V. Koltun, Deep global registration. in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2511–2520 (2020). https://doi.org/10.1109/CVPR42600.2020.00259
33. P. Babin, P. Giguère, F. Pomerleau, Analysis of robust functions for registration algorithms. in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1451–1457 (2019). https://doi.org/10.1109/ICRA.2019.8793791
34. M. Magnusson, A. Lilienthal, T. Duckett, Scan registration for autonomous mining vehicles using 3d-ndt. J. Field Robot. **24**(10), 803–827 (2007). https://doi.org/10.1002/rob.20204

35. N. Mellado, D. Aiger, N.J. Mitra, Super 4pcs fast global pointcloud registration via smart indexing. Comput. Graph. Forum **33**(5), 205–215 (2014). https://doi.org/10.1111/cgf.12446
36. Z. Chen, K. Sun, F. Yang, W. Tao, Sc2-pcr: A second order spatial compatibility for efficient and robust point cloud registration. in *CVPR*, pp. 13211–13221 (2022). https://doi.org/10.1109/CVPR52688.2022.01287
37. R.R. Hagege, J.M. Francos, Universal manifold embedding for geometrically deformed functions. IEEE Trans. Inf. Theory **62**(6), 3676–3684 (2016)
38. R. Sharon, J.M. Francos, R.R. Hagege, Geometry and radiometry invariant matched manifold detection. IEEE Trans. Image Process. **26**(9), 4363–4377 (2017)
39. Z. Yavo, Y. Haitman, J.M. Francos, L.L. Scharf, Matched manifold detection for group-invariant registration and classification of images. IEEE Trans. Signal Process. **69**, 4162–4176 (2021)
40. N. Lang, J.M. Francos, Deepume: Learning the universal manifold embedding for robust point cloud registration. in *British Machine Vision Conference (BMVC21)* (2021)
41. Y. Haitman, J.M. Francos, L.L. Scharf, Grassmannian dimensionality reduction for optimized universal manifold embedding representation of 3d point clouds. in *Proceedings of the IEEE Conference on Computer Vision*, pp. 4213–4221 (2021)
42. Y. Haitman, J.M. Francos, L.L. Scharf, Grassmannian dimensionality reduction using triplet margin loss for ume classification of 3d point clouds. in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (2022)
43. C. Mazzotti, N. Sancisi, V. Parenti-Castelli, A measure of the distance between two rigid-body poses based on the use of platonic solids, in *ROMANSY 21 - Robot Design, Dynamics and Control*. ed. by V. Parenti-Castelli, W. Schiehlen (Springer, Cham, 2016), pp.81–89
44. B.K. Horn, Closed-form solution of absolute orientation using unit quaternions. J. Opt. Soc. Am. A **4**(4), 629–642 (1987). https://doi.org/10.1364/JOSAA.4.000629
45. J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, J. Gall, SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. in *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)* (2019)
46. Q.-Y. Zhou, J. Park, V. Koltun, Open3D: A modern library for 3D data processing. arXiv:1801.09847 (2018)

Publisher's Note