ISPRS Journal of Photogrammetry and Remote Sensing 144 (2018) 61–79

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Hierarchical registration of unordered TLS point clouds based on binary shape context descriptor

Zhen Dong[a], Bisheng Yang[a,*], Fuxun Liang[a], Ronggang Huang[b], Sebastian Scherer[c,*]

[a] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China
[b] Institute of Geodesy and Geophysics, Chinese Academy of Sciences, Wuhan 430079, China
[c] The Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA

ARTICLE INFO

Keywords:
Point cloud registration
Binary shape context
Vector of locally aggregated descriptors
Point cloud similarity
Hierarchical registration
Multiple overlaps

ABSTRACT

Automatic registration of unordered point clouds collected by the terrestrial laser scanner (TLS) is the prerequisite for many applications including 3D model reconstruction, cultural heritage management, forest structure assessment, landslide monitoring, and solar energy analysis. However, most of the existing point cloud registration methods still suffer from some limitations. On one hand, most of them are considerable time-consuming and high computational complexity due to the exhaustive pairwise search for recovering the underlying overlaps, which makes them infeasible for the registration of large-scale point clouds. On the other hand, most of them only leverage pairwise overlaps and rarely use the overlaps between multiple point clouds, resulting in difficulty dealing with point clouds with limited overlaps. To overcome these limitations, this paper presents a Hierarchical Merging based Multiview Registration (HMMR) algorithm to align unordered point clouds from various scenes. First, the multi-level descriptors (i.e., local descriptor: Binary Shape Context (BSC) and global descriptor: Vector of Locally Aggregated Descriptor (VLAD)) are calculated. Second, the point clouds overlapping (adjacent) graph is efficiently constructed by leveraging the similarity between their corresponding VLAD vectors. Finally, the proposed method hierarchically registers multiple point clouds by iteratively performing optimal registration point clouds calculation, BSC descriptor based pairwise registration and point cloud groups overlapping (adjacent) graph update, until all the point clouds are aligned into a common coordinate reference. Comprehensive experiments demonstrate that the proposed algorithm obtains good performance in terms of successful registration rate, rotation error, translation error, and runtime, and outperformed the state-of-the-art approaches.

1. Introduction

Point clouds acquired by terrestrial laser scanners (TLS) are widely utilized for various applications such as 3D model reconstruction (Jung et al., 2014; Oesau et al., 2014; Xu et al., 2017a), cultural heritage management (Montuori et al., 2014; Yang and Zang, 2014), forest structure assessment (Liang et al., 2012; Kelbe et al., 2016a, 2016b), landslide monitoring (Prokop and Panholzer, 2009; Vosselman and Maas, 2010), and solar energy analysis (Huang et al., 2017). Since TLS has a limited field of view, multiple scans from different viewpoints are usually necessary to fully cover the geometry of a large-scale scene. A prerequisite for further processing is to register all individual scans in a common coordinate system, to obtain one large point cloud of the complete scene (Theiler et al., 2015). There are four key challenges for unordered point clouds registration: (1) uneven point densities of the point clouds caused by the mechanism of the TLS system (Zai et al., 2017), (2) the huge amount of data (millions or billions of points, hundreds of scans), which calls for computationally efficient techniques (Theiler et al., 2014), and (3) repetitive, symmetric, and incomplete structures (Theiler et al., 2015). (4) Limited overlaps and unknown relative position between multiview point clouds (Huber and Hebert, 2003). To address these challenges, extensive studies have been done to improve the accuracy, efficiency, and robustness of point cloud registration, which can be roughly categorized into pairwise and multiview registration according to the number of input point clouds (Huber and Hebert, 2003). Both of pairwise and multiview registration involve two steps: coarse and fine registration (Guo et al., 2013). The aim of coarse registration is to estimate an initial transformation between adjacent point clouds. The initial transformation is then further refined using a fine registration algorithm (e.g., iterative closest point (ICP) algorithm and its variants (Besl and McKay, 1992; Yang et al, 2013)). The scope of this paper will focus on pairwise and multiview coarse registration.

[*] Corresponding authors.
E-mail addresses: bshyang@whu.edu.cn (B. Yang), basti@andrew.cmu.edu (S. Scherer).

https://doi.org/10.1016/j.isprsjprs.2018.06.018
Received 22 December 2017; Received in revised form 22 June 2018; Accepted 26 June 2018

1.1. Pairwise coarse registration

A common pairwise coarse registration method should solve two major issues: the extraction of geometric features (e.g., points, lines, and planes) and the identification of corresponding feature pairs (Habib et al., 2010). Many studies using line or plane features to register point clouds have been widely reported. Stamos and Leordeanu (2003) proposed an autonomous registration method based on line features, which extracted the intersection lines of neighboring planes and calculated the transformation between adjacent scans using two or more corresponding line pairs. Dold and Brenner (2006) presented a registration method based on plane patches, which calculates rotation and translation parameters successively using at least three corresponding planar patches. Rabbani et al. (2007) first detected the geometric primitives (i.e., planes, spheres, cylinders and tori) from each individual scan, then registered the scans of industrial sites by the corresponding geometric primitives. Theiler and Schindler (2012) first generated the virtual tie points by intersecting triples of detected planes, then virtual tie points are matched using their descriptors (e.g., the intersection angles between planes the extent of planar segments, and the smoothness of planes). Yang et al. (2016) first detected feature lines derived from pole-like objects and vertical planes then matched them using both geometrical constraints and their semantic information. Xu et al. (2017b) proposed an automatic marker-free solution for coarsely aligning two scans of point clouds, utilizing the geometric constraint formed by planar surfaces of building facades and ground surface. Although the feature line/plane-based methods work very well for urban scenes with numerous man-made objects, they suffer the difficulty in natural scenes (e.g., mountain, river, and forest) with less line/plane features.

In general, the point-based methods are more popular in pairwise coarse registration due to their feasibility to various scenes (Böhm and Becker, 2007; Barnea and Filin, 2008; Weinmann et al., 2011; Theiler et al., 2014; Weber et al., 2015). Most of the point-based registration methods typically contain four consecutive processes. First, the keypoint detectors (e.g., local surface patches (Chen and Bhanu, 2007), intrinsic shape signature (Zhong, 2009)), 2.5D SIFT (Lo and Siebert, 2009), 3D SURF (Knopp et al., 2010) and Harris 3D (Sipiran and Bustos, 2011)) are utilized to extract keypoints from raw point clouds. Second, the feature descriptors (e.g., spin image (Johnson and Hebert, 1999), 3D Shape Contexts (Frome et al., 2004), fast point feature histograms (FPFH) (Rusu et al., 2009), signature of histograms of orientations (Tombari et al., 2010), rotational projection statistics (RoPS) (Guo et al., 2013), and Tri-Spin-Image (Guo et al., 2015)) are calculated to encode spatial shape information around each keypoint. Third, the correspondences are determined through various feature matching strategies (e.g., reciprocal correspondence (Pajdla and Gool, 1995), correlation coefficient (Johnson and Hebert, 1999), and chi-square test (Zhong, 2009)). Finally, some robust transformation estimation algorithms (e.g., RANSAC (Fischler and Bolles, 1981), polygon-based correspondence rejector (Weber et al., 2015), geometric consistency constraints (Tombari et al., 2010; Yang et al., 2016)) are leveraged to eliminate incorrect correspondences and calculate the rigid transformation between adjacent point clouds based on the remaining correspondences.

There are some point-based methods in the literature which do not follow the aforementioned feature descriptors calculation and feature matching paradigm. For example, the 4-Points Congruent Sets (4PCS) (Aiger et al., 2008) and its variants (e.g., Keypoint-based 4PCS (K-4PCS) (Theiler et al., 2014), Geodesic distances-based 4PCS (GD-4PCS) (Ge, 2016), and semantic-keypoint-based 4PCS (SK-4PCS) (Ge, 2017)) determine the corresponding four-point base sets by exploiting the rule of intersection ratios instead of feature descriptor calculation and matching. These algorithms are more robust to a set of disturbances, including noise, varying point density, clutter, and occlusion.

Although the abovementioned pairwise registration methods can generally provide satisfactory point cloud registration results, they still have some common limitations. First, the existing 3D feature descriptors still suffer from low descriptiveness, and weak robustness (Dong et al., 2017). Second, most of the pairwise registration algorithms are time-consuming especially for large-scale point clouds with a huge amount of data.

1.2. Multiview coarse registration

For multiview coarse registration, both the overlaps and the relative poses are unknown, which makes the problem considerably harder than pairwise coarse registration (Huber and Hebert, 2003). The multiview coarse registration algorithms involve three interrelated tasks: (1) recovering the overlap information or the view order of input point clouds; (2) estimating the rigid transformations between each pair of overlapping point clouds (pairwise registration); (3) determining the absolute poses of each input point cloud, which is the ultimate goal of multiview coarse registration.

In the last decades, extensive studies on multiview coarse registration have been done using the minimum spanning tree based strategy (Huber and Hebert, 2003; Weber et al., 2015; Yang et al., 2016). Huber and Hebert (2003) first applied the spin image based pairwise coarse registration to all pairs of point clouds to construct a fully connected and weighted graph, where the overlap distance was calculated as the weight of each edge. They then generated the minimum spanning tree (MST) (Kruskal, 1956) of the graph to register multiview point clouds to a uniform coordinate reference. Following the similar workflow, Weber et al. (2015) and Yang et al. (2016) respectively used FPFH (Rusu et al., 2009), and semantic feature point based algorithms for exhaustive pairwise coarse registration to construct a fully connected graph, and generated a minimum spanning tree of the graph that maximizes the number of point correspondences. These algorithms avoid the issue of pose conflict by finding the minimal spanning tree that connects all nodes to the reference node, but do not exploit the redundant information provided by multiple edges to reduce registration error.

The global energy optimization based strategy (Theiler et al., 2015), and embedded confidence metrics based strategy (Kelbe et al., 2016b) are proposed to remove erroneous local matches and reduce propagation errors by exploiting the redundant information (e.g., loop constraints). Theiler et al. (2015) first utilized the K-4PCS based pairwise registration method (Theiler et al., 2014) to all pairs of input point clouds to get multiple putative transformations between each pair. They then constructed a graphical model and its energy function to include the loop consistency constraints, and disambiguated the pairwise transformations by minimizing the energy function using Lazy-Flipper algorithm (Andres et al., 2012). Kelbe et al. (2016b) leverage the pairwise embedded confidence metric (Kelbe et al., 2016a) to align exhaustive pairwise input point clouds. They then initialized a graph using the set of relative pairwise transformations and calculated the Dijkstra spanning trees of each node. Finally, to address potential inconsistent matches, the competing Dijkstra spanning trees were aligned into a common coordinate system, from which effective pose estimates and error estimates were calculated. One major limitation of these algorithms is their high computational complexity due to the expensive exhaustive search. For a set of N point clouds, the computational complexity of the minimum spanning tree based algorithms (Huber and Hebert, 2003; Weber et al., 2015; Yang et al., 2016), global energy optimization based algorithm (Theiler et al., 2015), and embedded confidence metrics based algorithm (Kelbe et al., 2016b) is $O(N^2)$ as they need to exhaustively register every pair of point clouds (i.e., C_N^2). These algorithms are therefore very time-consuming and infeasible for the registration of a large number of point clouds.

To reduce the computational complexity and improve the efficiency of multiview registration algorithms, extensive studies on shape growing based strategies have been done (Mian et al., 2006; Guo et al., 2014). Mian et al. (2006) constructed a connected graph by first choosing the point clouds with the maximum surface area as the root node and iteratively merged other point clouds with enough corresponding 3D tensors and passed a global verification to the graph. Following the similar workflow, Guo et al. (2014) started by initializing the search space with all the input point clouds, and then selected a point cloud from the search space as the seed shape. The shape iteratively grew by performing pairwise registration between itself and the remaining point clouds in the search space. This algorithm was more efficient compared to the minimum spanning tree based algorithms, and global energy optimization based algorithm. One major limitation of this algorithm is that point clouds covering a small part of an object or with limited overlaps cannot be registered (Guo et al., 2014).

Another trend of multiview registration depends on external sensors auxiliary (e.g., GPS, IMU, and smartphone) to record the position and orientation of each scan (Asai et al., 2005; Pu et al., 2014; Chen et al., 2017). For example, Chen et al. (2017) combined TLS with a low-cost smartphone for the multiview coarse registration. More precisely, the scanner positions are roughly measured by the smartphone GPS and the distances between neighboring scanner positions are used as the translation constraint in the multiview registration step. External sensors are helpful for registration of terrestrial point clouds, although the high cost, tedious data collection processes, and signal occlusion of external sensors limit the application of these methods.

Matching between images and point clouds share several common research problems, therefore the related work from the structure from motion (SfM) is also briefly introduced. SfM algorithms take a set of images as input and produce the camera parameters and a set of 3D points visible in the images (Furukawa and Hernández, 2015). The development of high-quality feature detectors (Alahi et al., 2012; Bay et al., 2008a, 2008b; Rosten and Drummond, 2006; Leutenegger et al., 2011), which enable build longer and higher quality tracks from images, is a crucial development to make the SfM work with unstructured datasets. Another development to tackle large-scale SfM of unstructured photo collections is to accelerate the matching stage by the efficient indexing (Nister and Stewenius, 2006), the simplified connectivity graph of the tracks (Snavely et al., 2008) and parallelization (Agarwal et al., 2011; Frahm et al., 2010).

1.3. Contributions

Although the reported methods can generally provide satisfactory registration results, they still have limitations. One major limitation of these algorithms is their considerable time cost and high computational complexity due to the time-intensive feature matching process of pairwise registration and exhaustive pairwise search of multiview registration, which result in infeasible for the registration of a large number of point clouds. Another one is that most of the existing algorithms only leverage pairwise overlaps and rarely use the overlaps between multiple point clouds, therefore these methods may have difficulty dealing with point clouds with limited pairwise overlaps. To overcome these limitations, this paper presents a **H**ierarchical **M**erging based **M**ultiview **R**egistration (HMMR) algorithm to align unordered point clouds from various scenes and validates its performance on six challenging datasets. The main contributions of the proposed method are as follows.

- We propose an efficient algorithm to construct point clouds overlapping (adjacent) graph by calculating the similarity between their corresponding Vector of Locally Aggregated Descriptor (VLAD) vectors instead of exhaustive pairwise registration, which significantly reduces the computational complexity of multiview registration.
- We introduce a robust and efficient pairwise point cloud registration algorithm based on Binary Shape Context (BSC) descriptor (Dong et al., 2017). On the one hand, the BSC descriptor is robust to noise, varying point density, occlusion and clutter, which can be used for registration of point clouds from various scenes. On the other hand, the similarity between BSC descriptors can be measured by the Hamming distance, which makes the feature matching process extremely fast.
- We present a hierarchical merging based registration algorithm, which handles a marker-free and multiview registration without any prior knowledge about the view orders or position of the point clouds. There are two important features of the proposed algorithm. On the one hand, the optimal point clouds merging order is automatically determined according to the similarity between point clouds, resulting in the improvement of registration accuracy. On the other hand, overlaps between multiple point clouds (as opposed to just pairwise overlaps) is leveraged during point clouds merging process, which enhances the capability of handling point clouds with limited overlaps.

The rest of this paper is organized as follows. Following this introduction, Section 2 gives a detailed description of the proposed point cloud registration method. Then the proposed method is validated in experimental studies in Section 3. Finally, the conclusions and future research directions are presented in Section 4.

2. Methodology

In this section, we propose a **H**ierarchical **M**erging based **M**ultiview **R**egistration (HMMR) algorithm with a successive scheme that includes multi-level feature calculation (Section 2.2), point clouds overlapping (adjacent) graph construction (Section 2.3), and hierarchical registration of point clouds (Section 2.4).

2.1. Hierarchical registration framework

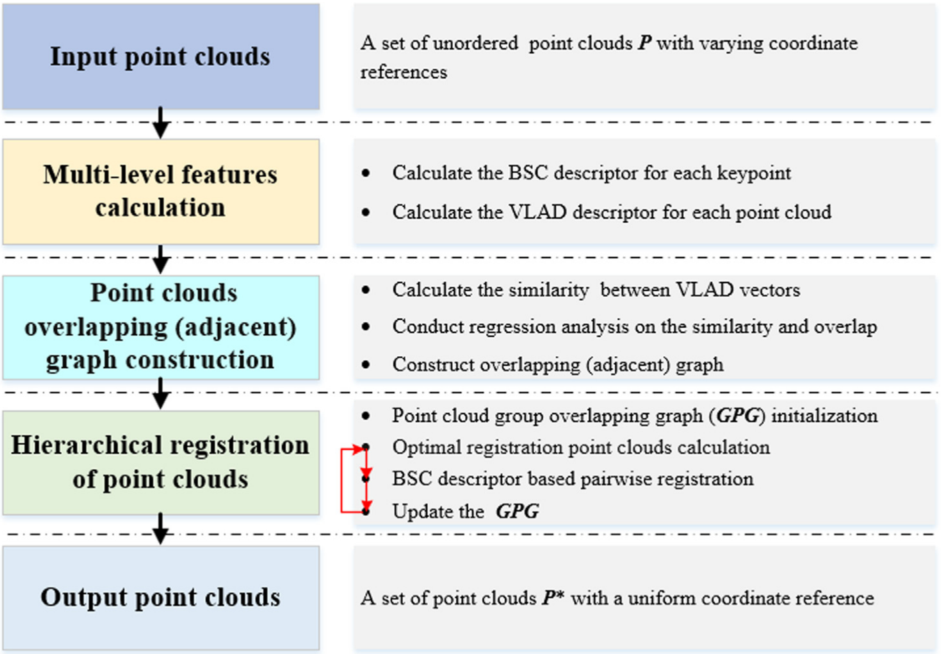
The task of the proposed method is to align a set of unordered point clouds into a uniform coordinate reference. It's worth noting that we neither make assumptions about the position and view order of the sensors, nor the overlap between collected point clouds, meaning that consecutive views do not imply spatial adjacency. Fig. 1 shows an overview of the proposed hierarchical registration framework. The input of the framework is a set of unordered point clouds $P\{P_1, \cdots P_i, \cdots P_N\}$ with varying coordinate references, where P_i represents the ith point cloud, and N is the total number of input point clouds. First, the local descriptor (i.e., BSC) of each keypoint and the global descriptor (i.e., VLAD) of each point cloud are calculated (Section 2.2). Second, the point clouds overlapping (adjacent) graph is efficiently constructed by leveraging the similarity between their corresponding VLAD vectors (Section 2.3). Third, the proposed method hierarchically registers the point clouds by iteratively performing three successive processes (i.e., optimal registration point clouds calculation, BSC descriptor based pairwise registration, and point cloud groups overlapping (adjacent) graph update) until all the point clouds are aligned into a common coordinate reference (Section 2.4). The final output of the proposed method is the set of point clouds $P^*\{P_1^*, \cdots P_i^*, \cdots P_N^*\}$ with a uniform coordinate reference, where P_i^* represents the transformed point cloud of P_i.

2.2. Multi-level feature calculation

In this section, the local BSC descriptor and the global VLAD (Vector of Locally Aggregated Descriptor) descriptor are calculated, which will be used for the following processes of pairwise registration and overlapping point cloud pairs estimation respectively.

2.2.1. Binary shape context calculation

In this section, our previously developed local BSC descriptor (Dong et al., 2017) is calculated to encode spatial shape information around each keypoint. To make the description of the complete system self-contained, we start with a brief description of the BSC descriptor, as shown in Fig. 2.



Input point clouds

A set of unordered point clouds *P* with varying coordinate references

Multi-level features calculation

- Calculate the BSC descriptor for each keypoint
- Calculate the VLAD descriptor for each point cloud

Point clouds overlapping (adjacent) graph construction

- Calculate the similarity between VLAD vectors
- Conduct regression analysis on the similarity and overlap
- Construct overlapping (adjacent) graph

Hierarchical registration of point clouds

- Point cloud group overlapping graph (*GPG*) initialization
- Optimal registration point clouds calculation
- BSC descriptor based pairwise registration
- Update the *GPG*

Output point clouds

A set of point clouds *P** with a uniform coordinate reference

Fig. 1. Overview of the proposed point cloud registration framework.

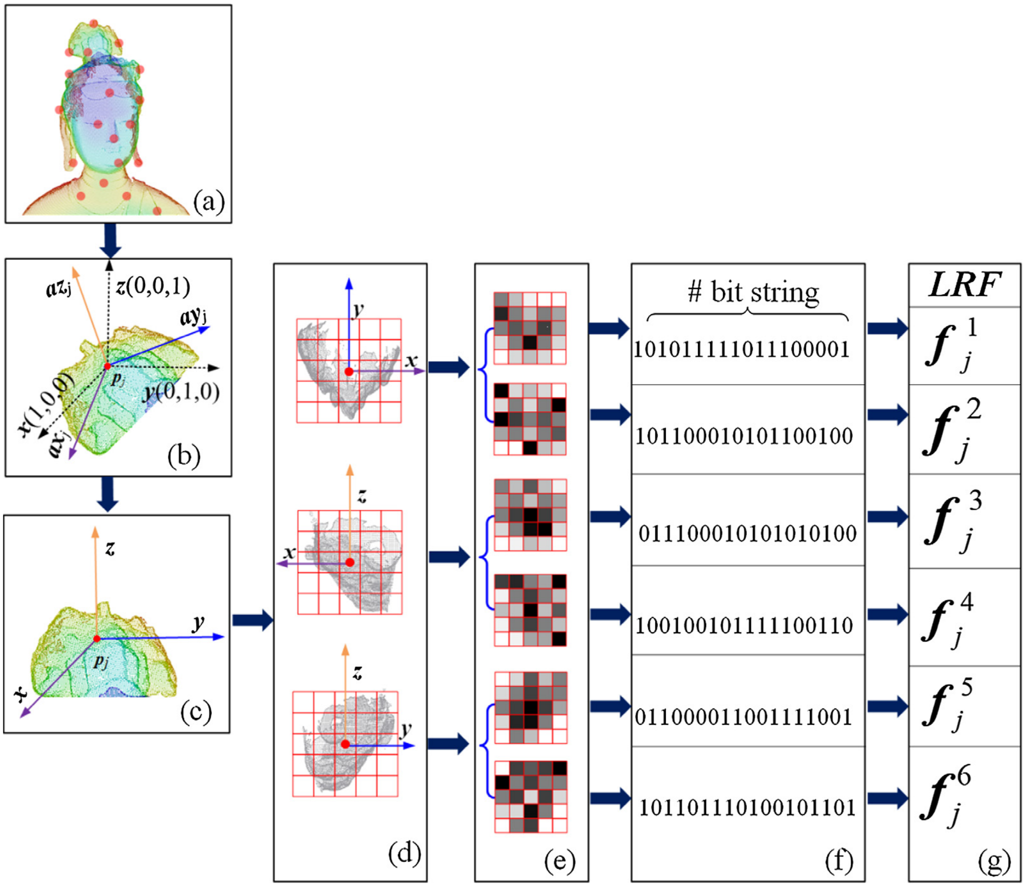


Fig. 2. An illustration of the generation of a BSC descriptor: (a) the input point cloud and detected keypoints (i.e., the red points), (b) the neighboring points of keypoint p_j and the constructed LRF, (c) the transformed neighboring points with respect to the LRF, (d) the projected neighboring points with respect to three projection planes respectively, (e) the projection point density feature and projection point distance feature corresponding to each projection plane, (f) the corresponding bit strings generated by feature difference test, and (g) the generated BSC descriptor by combining the LRF and the six bit strings. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

First, the keypoints of each point cloud are detected by the keypoint detector proposed by Mian et al. (2010), as shown in Fig. 2a. More specifically, the algorithm detects keypoints at different pre-selected scales and selects the scale that maximizes the ratio between the first two principle axes as the inherent scale. Both the location and inherent scale $r^{(\text{scale})}$ (i.e., supporting radius) of a keypoint are determined in the detection phase, which will be used to construct the BSC descriptor in the subsequent phase.

Second, the Local Reference Framework (LRF) at each keypoint is constructed and its neighboring points are then transformed with respect to the constructed LRF, as shown in Fig. 2b and c. The details of the LRF construction algorithm are as follows: (1) Construct the covariance matrix $\boldsymbol{M}$ using p_j and its neighboring points within a supporting radius $r^{(\text{scale})}$, which is automatically determined in the keypoint-detection step. (2) Compute the eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$ of $\boldsymbol{M}$ in decreasing order of magnitude and the corresponding eigenvectors $\{e_1, e_2, e_3\}$ by performing an eigenvalue decomposition of $\boldsymbol{M}$. (3) Define the LRF by adopting p_j as the origin and e_1, e_2, $e_1 \otimes e_2$ as the x-, y- and z-axes, respectively, where $\otimes$ represents the cross product of vectors.

Third, the transformed neighboring points are projected onto three coordinate planes (i.e., the xy-, xz- and yz-planes) and subsequently divided into bins, as shown in Fig. 2d; then the projection density and distance features of each bin are calculated and normalized into [0, 255], as shown in Fig. 2e.

Fourth, the projection density and distance features are further transformed to be binary strings by the difference test, as shown in Fig. 2f. More specifically, the difference tests of each kind feature (i.e., projection density and projection distance) on each projection plane (i.e., the xy-, xz- and yz-planes) choose g feature pairs based on the predefined rules and calculate their difference respectively. The value of one difference test is set as 1 if there is a significant difference between the selected feature pair, otherwise it is set as 0, as Eq. (1).

$$\tau(f_{b_l}, f_{b_l'}) = \begin{cases} 1 & if \ |f_{b_l} - f_{b_l'}| > \sigma_T \\ 0 & else, \end{cases} \tag{1}$$

where | | represents the absolute value, $f_{b_l}, f_{b_l'}$ is one of the selected feature pairs, $\tau(f_{b_l}, f_{b_l'})$ denotes the difference test of f_{b_l} and $f_{b_l'}$, σ_T is a predefined threshold.

Finally, the BSC descriptor of each keypoint p_j is generated by combining the LRF with the six corresponding bit strings into a feature vector b_j, as Eq. (2).

$$b_j(p_j, ax_j, ay_j, az_j, f_j) \tag{2}$$

where p_j, ax_j, ay_j, az_j are the origin, the x-axis, y-axis, and z-axis of the local coordinate reference, f_j is the six corresponding bit strings. The readers can refer to Dong et al. (2017) for more details on the BSC descriptors extraction.

2.2.2. Vector of locally aggregated descriptor calculation

After BSC descriptors calculation, the proposed method groups all the BSC descriptors into K clusters using k-means clustering algorithm (Jégou et al., 2010), and regards the K cluster centers as the visual words. Then the local BSC descriptors derived from each individual point cloud and the K visual words are received as the input of Algorithm 1 to calculate the VLAD vector of the point cloud. The pseudocode of the VLAD vector calculation is detailed in Algorithm 1.

Algorithm 1: VLAD calculation for point cloud P_i

```
    Notation:
    $B_i\{b_1^i, \cdots b_j^i, \cdots b_{T_i}^i\}$: the set of BSC descriptors derived from point cloud $P_i$, where $T_i$ is the number of descriptors;
    $U\{\mu_1, \mu_2, \cdots \mu_K\}$: the set of visual words, where $K$ is number of visual words;
    $d$: the dimension of BSC descriptors;
    $V_i$: the VLAD vector of point cloud $P_i$(the dimension of VLAD is $K*d$).
  Input: $B_i\{b_1^i, \cdots b_j^i, \cdots b_{T_i}^i\}$ and $U\{\mu_1, \mu_2, \cdots \mu_K\}$
  Output: $V_i$
  % initialization
1   for $k = 1, 2, \cdots, K$
2     $V_i(k) \leftarrow 0_d$, $0_d$ represents a $d$-dimensional zero vector;
3   End for
  % difference accumulation
4   for $j = 1, 2, \cdots, T_i$
5     $k = \arg\min_{n=1,2\cdots K} \|b_j^i - \mu_n\|$, $\|b_j^i - \mu_n\|$ is the L1-norm between $b_j^i$ and $\mu_n$
6     $V_i(k) \leftarrow V_i(k) + b_j^i - \mu_k$
7   End for
  % concatenation and normalization
8   $V_i = [V_i(1), \cdots V_i(k) \cdots V_i(K)]$
9   $V_i \leftarrow \frac{V_i}{\|V_i\|_2}$, $\|V_i\|_2$ represents the L2-norm of $V_i$.
```

The algorithm first accumulates the differences $b_j - \mu_k$ of each BSC descriptor b_j assigned to its nearest visual word μ_k (i.e., step 4–7 of Algorithm 1). Then the Kd-dimensional VLAD vector is acquired by concatenating the K difference accumulation $V_i(k)$ and subsequently L2-normalized (i.e., step 8–9 of Algorithm 1), where $V_i(k)$ is a d-dimensional vector accumulating the difference between visual word μ_k and its corresponding BSC descriptors. The readers can refer to Jégou et al. (2010, 2012) for more details on the VLAD calculation for image search and classification.

2.3. Point clouds overlapping (adjacent) graph construction

For multiview registration, both the overlaps and the relative position among point clouds are unknown, which make the problem considerably harder. To recover the underlying overlaps and the adjacent relation, most of the existing algorithms first perform the exhaustive pairwise coarse registration then determine the reliable point cloud pairs by a predefined metric, which is extraordinarily time-consuming and high-complexity. Before multiview registration, we efficiently recover the underlying adjacent relation or overlaps between point clouds by calculating the similarity between their corresponding VLAD vectors, thus significantly reducing the computational complexity compared with exhaustive pairwise registration.

2.3.1. VLAD similarity calculation

Intuitively, point clouds with larger overlap are expected to have more similar VLAD vectors. The most similar and dissimilar components of the VLAD vectors are associated with the BSC descriptors located in the overlapping areas, and nonoverlapping areas respectively. Starting from this consideration, the similarity $\text{sim}(V_i, V_j)$ between two

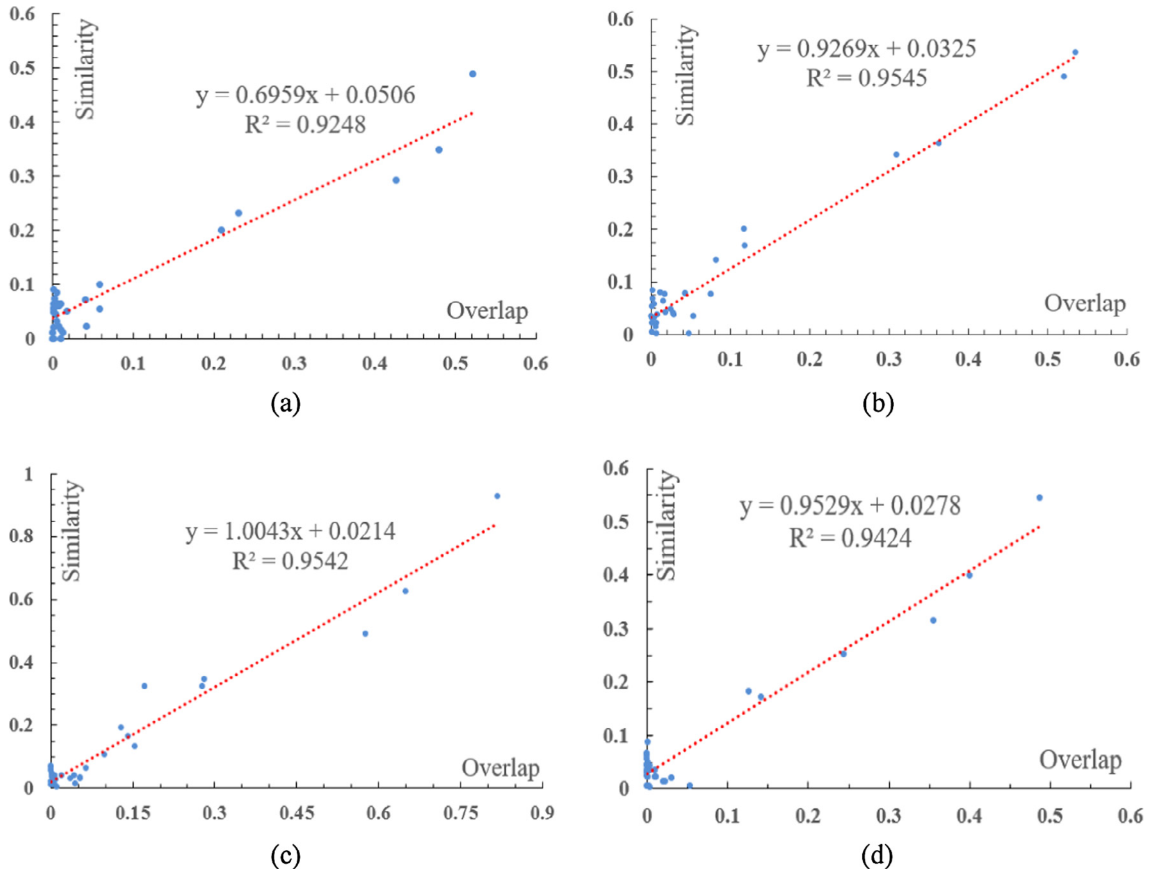


Fig. 3. The correlation between similarity and overlap: (a) point cloud pairs formed by 2*nd* point cloud and the remaining point clouds, (b) point cloud pairs formed by 4*th* point cloud and the remaining point clouds, (c) point cloud pairs formed by 16*th* point cloud and the remaining point clouds, and (d) point cloud pairs formed by 25*th* point cloud and the remaining point clouds.

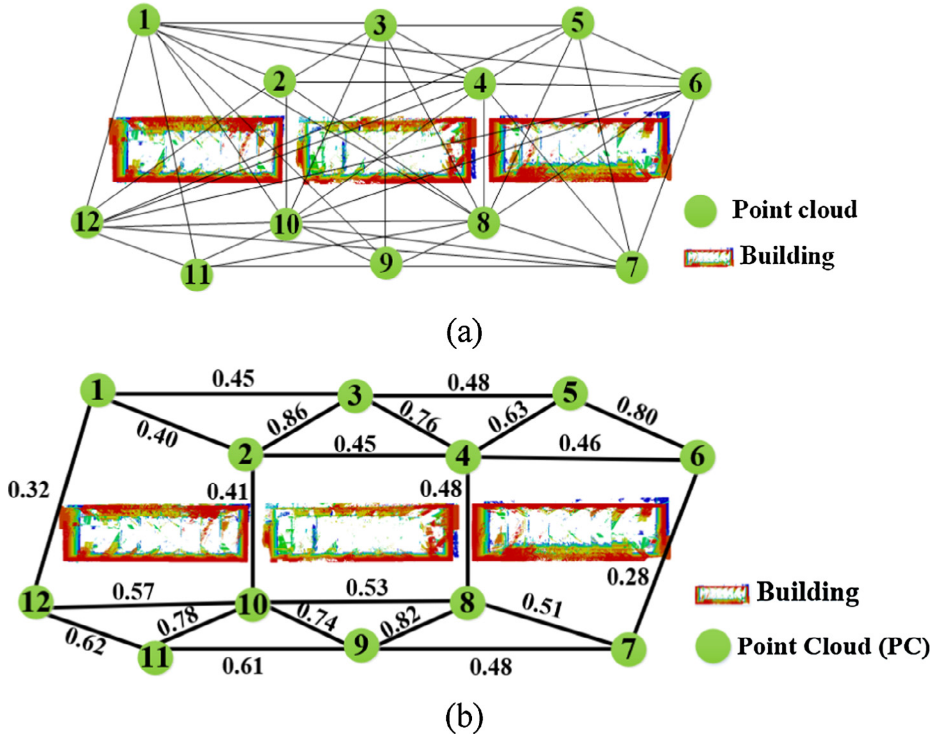


Fig. 4. A comparison of the proposed algorithm with exhaustive pairwise search: (a) the generated graph by exhaustive pairwise search, and (b) the generated overlapping (adjacent) graph by the proposed algorithm.

VLAD vectors V_i, and V_j is defined as a weighted distance:

$$\mathrm{sim}(V_i, V_j) = \frac{1}{K}\sum_{k=1,2\cdots K} w_k \frac{1}{\|V_i(k)-V_j(k)\|} \tag{3}$$

where $\|V_i(k)-V_j(k)\|$ is the L1-norm of $V_i(k)$ and $V_j(k)$, $\frac{1}{\|V_i(k)-V_j(k)\|}$ represents the similarity of $V_i(k)$ and $V_j(k)$, w_k is the weight of $\frac{1}{\|V_i(k)-V_j(k)\|}$. The pairs $\{(V_i(1), V_j(1)), \cdots(V_i(k), V_j(k)), \cdots(V_i(K), V_j(K))\}$ are sorted in descending order with respect to their similarity $\frac{1}{\|V_i(k)-V_j(k)\|}$. The weight $w_k = 1-\frac{k}{K}$ amplifies the influence of the more similar pairs and reduces the impact of the more dissimilar pairs.

2.3.2. Regression analysis of similarity and overlap

The correlation between VLAD similarity and point cloud overlap is validated on the park dataset (refer to Section 3.1.1) using the linear regression analysis technology (Montgomery and Peck, 2007). Given two point clouds P_i, P_j and their corresponding VLAD vectors V_i and V_j, the similarity $\mathrm{sim}(V_i, V_j)$ is calculated as Eq. (3), and the overlap between P_i and P_j is calculated as Eq. (4).

$$\mathrm{overlap}(P_i, P_j) = \frac{2{*}\text{point number in overlapping areas}}{\text{the total point number in both point clouds}} \tag{4}$$

The 2nd, 4th, 16th and 25th point clouds of Park dataset are randomly selected to validate the correlation between VLAD similarity and point cloud overlap. The readers can refer to Section 3.1.1 for more details about the Park dataset. Fig. 3 shows the strong positive correlation between VLAD similarity (normalized to [0, 1]) and point cloud overlap, with the vary R-squared (R^2) between 0.92 and 0.95.

2.3.3. Overlapping (adjacent) graph construction

We formulate the overlapping (adjacent) relations between point clouds as a graph, where each point cloud is a node and edges encode similarity (overlap) between pairs of point clouds. For each given point cloud P_i, we choose its top q similar point clouds as its overlapping (adjacent) point clouds and connect them to P_i with edges. Fig. 4 illustrates a simple case of the constructed overlapping (adjacent) graph, where the black numbers on the edges indicate the similarity of point clouds. Note that q is set as 3 in Fig. 4, the optimal value of q will be discussed in the Section 3.1.3. Fig. 4a and b respectively show the generated graphs of exhaustive pairwise search and the proposed method for the 12 input point clouds. It is worth noting that, most of the existing algorithms perform the exhaustive pairwise coarse registration to recover the underlying overlaps or the adjacent relation, which is extraordinarily time-consuming and high-complexity. The proposed method efficiently recovers the underlying adjacent relation or overlaps between point clouds by leveraging the similarity between their corresponding VLAD vectors. The constructed overlapping (adjacent) graph will be used for the following hierarchical registration process.

2.4. Hierarchical registration of point clouds

Given a set of input point clouds $P\{P_1, \cdots P_i, \cdots P_N\}$ and the constructed point cloud overlapping (adjacent) graph GP, the proposed algorithm hierarchically registers them into a uniform coordinate reference according to Algorithm 2, illustrated in Fig. 5.

Algorithm 2: Hierarchical registration of point clouds

```
     Notation:
       $P\{P_1, \cdots P_i, \cdots P_N\}$: the set of input point clouds;
       $B\{B_1, \cdots B_i, \cdots B_N\}$: the BSC descriptor set of input point clouds;
       $PG\{PG_1, \cdots PG_i, \cdots PG_N\}$: the set of initialized point cloud groups;
       $GP$: the point clouds overlapping (adjacent) graph;
       $GPG$: the point cloud groups overlapping (adjacent) graph;
     Input: $P\{P_1, \cdots P_i, \cdots P_N\}$ and $GP$
     Output: $P^*\{P_1^*, \cdots P_i^*, \cdots P_N^*\}$ the set of registered point clouds
     % Point cloud groups initialization
1      for $i = 1, 2, \cdots, N$
2        $PG_i \leftarrow P_i$;
3      End for
4        $GPG \leftarrow GP$;
     % Optimal registration point clouds calculation
5      Select the two most similar point cloud groups $PG_s$ and $PG_t$ from $GPG$;
6      Further select the overlapping point clouds between $PG_s$ and $PG_t$ based on $GP$ as the pairwise registration point clouds $RP_s$ and $RP_t$;
     % BSC descriptor based pairwise registration
7      Estimate the feature correspondence set $FC\{c_1, c_2, \cdots c_{MFC}\}$ between $RP_s$ and $RP_t$;
8      Eliminate the incorrect correspondences from $FC$ to get $GC\{c_1, c_2, \cdots c_{MGC}\}$;
9      Calculate the coarse transformation $T_{s,t}^C$ based on $GC$ by Eq. (7);
10     Calculate the fine transformation $T_{s,t}^F$ by the pairwise ICP on point clouds $RP_s$ and $RP_t$;
11     Transform the point clouds in $PG_t$ using $T_{s,t}^F$ to get $PG_t^*$;
     % Point cloud groups overlapping (adjacent) graph $GPG$ update
12     Merge point cloud group $PG_s$ and $PG_t^*$ into $PG_{s,t}$ in the graph $GPG$;
13     Merge BSC descriptor set $B_s$ and $B_t^*$ into $B_{s,t}$;
14     Calculate the VLAD vector $V_{s,t}$ of $PG_{s,t}$ based on $B_{s,t}$ by Algorithm 1;
15     Update the similarity between $PG_{s,t}$ and its conterminal point cloud groups in $GPG$;
     % Iteration
16     Repeat steps 5–15, until all the input point clouds are registered into a uniform coordinate reference;
     % Multiview ICP
17     Refine all the registered point clouds by using the multiview ICP algorithm (Williams and Bennamoun, 2001) on them to eliminate the error propagation;
```

2.4.1. Point cloud groups initialization

The proposed method first initializes the set of point cloud groups $PG\{PG_1, \cdots PG_i, \cdots PG_N\}$ by assigning each point cloud P_i as a point cloud group PG_i, which is defined as a group of point clouds with a uniform coordinate reference; then assign the point cloud overlapping (adjacent) graph GP to point cloud groups overlapping (adjacent) graph

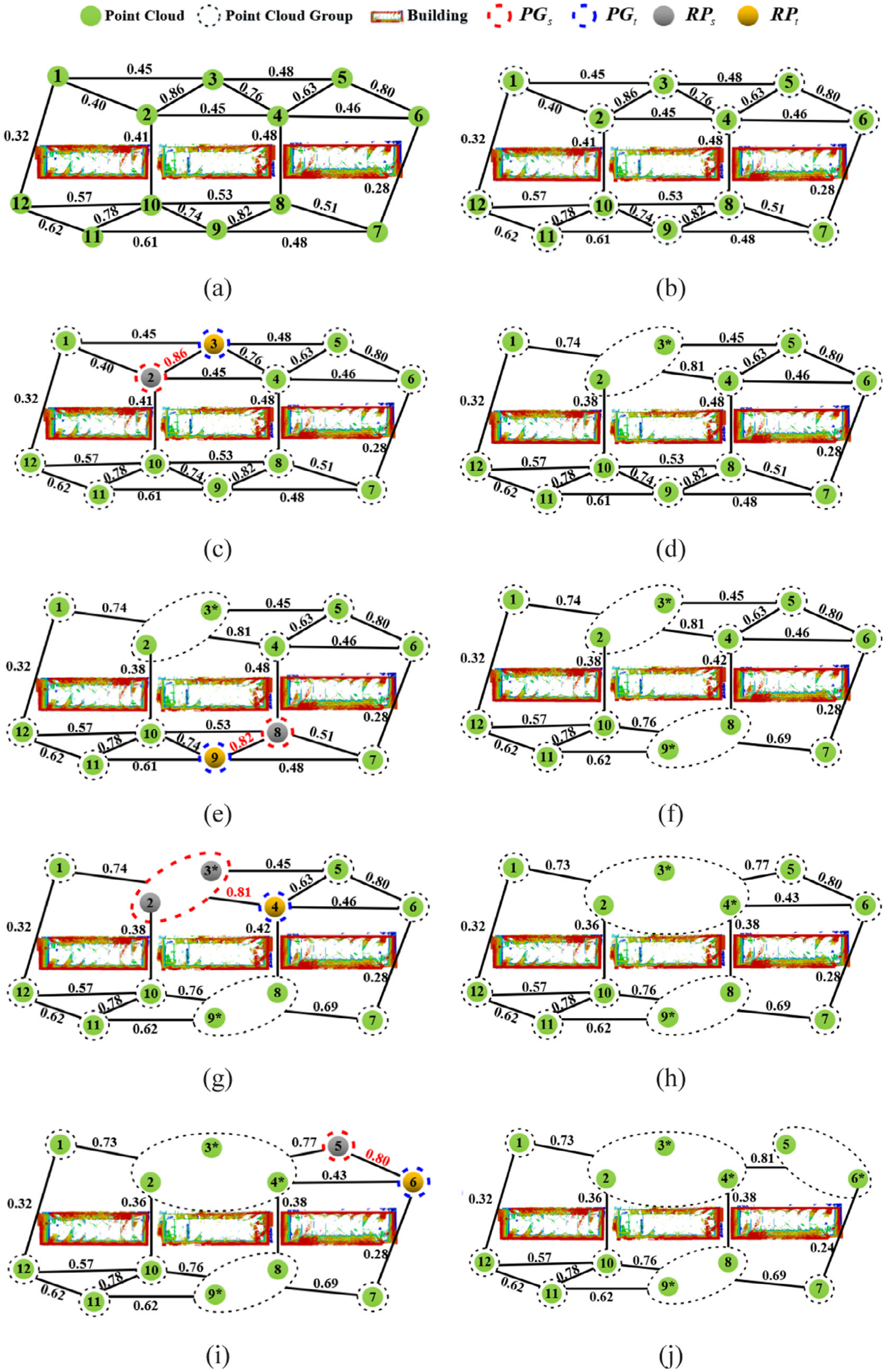


Fig. 5. An illustration of hierarchical registration: (a) the input point cloud overlapping (adjacent) graph, (b) the initialized point cloud groups overlapping (adjacent) graph, (c) and (d) the first iteration, (e) and (f) the second iteration, (g) and (h) the third iteration, (i) and (j) the fourth iteration, (k) and (m) the fifth iteration, (n) and (p) the last iteration.

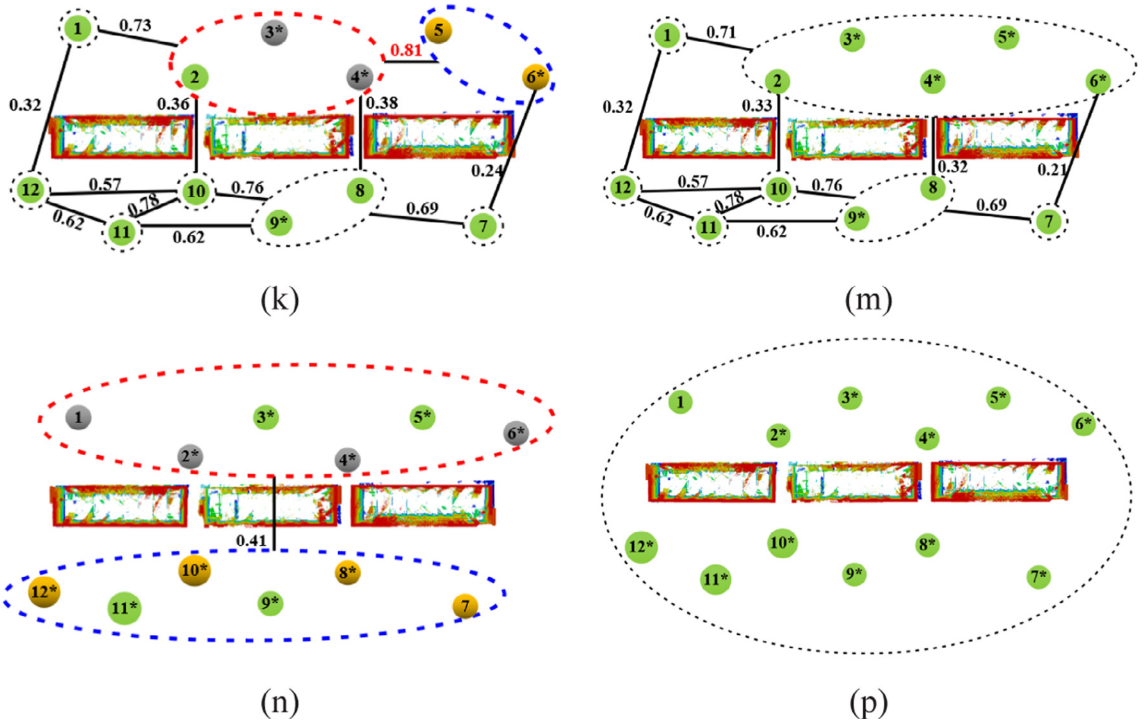


Fig. 5. (*continued*)

Table 1
The detailed description of the six datasets.

Dataset	Scanner	#Scans	#Pts	Dimensions (m)	Average point density (points/m^2)
Park	VZ-400	32	0.28 billion	1450 * 650	442
Campus[1]	VZ-400	132	2.75 billion	1800 * 800	206
Mountain	ScanStation C5	22	0.13 billion	900 * 120	154
Tunnel	VZ-400	26	0.99 billion	400 * 60	3366
Bridge	FocusS 150	29	0.73 billion	400 * 400	273
River	VZ-400	13	0.23 billion	1500 * 400	190

[1] The campus dataset was download from http://kos.informatik.uni-osnabrueck.de/3Dscans/.

GPG. Fig. 5a and b shows the constructed point cloud overlapping (adjacent) graph *GP* and the initialized point cloud groups overlapping (adjacent) graph *GPG*, where the black numbers on the edges indicate the similarity of the nodes.

2.4.2. Optimal registration point clouds calculation

The proposed method first selects the source point cloud group PG_s and target point cloud group PG_t with most VLAD similarity from point cloud groups overlapping graph *GPG*, then further selects the overlapping point clouds between PG_s and PG_t (i.e., the point clouds connected by edges in the point cloud overlapping graph *GP*, Fig. 5a) as the pairwise registration point clouds RP_s and RP_t.

Fig. 5c–p illustrates the hierarchical registration in the first 5 and the last iterations. Note that, the point cloud groups will become larger and larger during the iterations, hence direct registration of point cloud groups PG_s and PG_t will become dramatically time-consuming. To improve the efficiency of registration, especially for point cloud groups containing a large number of point clouds, we register the point cloud groups by only utilizing their overlapping point clouds instead of all the point clouds in the groups. For example, in the last iteration, the proposed method first selects the source and target point cloud groups $PG_s\{P_1, P_2^*, P_3^*, P_4^*, P_5^*, P_6^*\}$ and $PG_t\{P_7, P_8^*, P_9^*, P_{10}^*, P_{11}^*, P_{12}^*\}$ from point cloud groups overlapping graph *GPG*, then further selects the point clouds $\{P_1, P_2^*, P_4^*, P_6^*\}$ from PG_s and the point clouds $\{P_7, P_8^*, P_{10}^*, P_{12}^*\}$ from PG_t as the pairwise registration point clouds RP_s and RP_t (point clouds (P_1, P_{12}), (P_2, P_{10}), (P_4, P_8), (P_6, P_7) are connected by edges in Fig. 5a), shown as the gray and brown points respectively in Fig. 5n. Only the point clouds RP_s and RP_t will be used for the following pairwise registration, which significantly improves the efficiency of pairwise registration.

2.4.3. BSC descriptor based pairwise registration

The proposed algorithm aligns RP_s to RP_t with a successive scheme that includes correspondence estimation, incorrect correspondences rejection, coarse transformation calculation, and fine transformation calculation.

- **correspondence estimation**

Let $B_s\{b_1^s, b_2^s, \cdots b_{T_s}^s\}$ and $B_t\{b_1^t, b_2^t, \cdots b_{T_t}^t\}$ respectively be the sets of BSC descriptors from RP_s and RP_t. For a descriptor b_k^s in B_s, if there exists a descriptor b_h^t in B_t fulfilling Eq. (5), b_k^s and b_h^t are considered as a correspondence.

$$\begin{cases} h = \arg \min\limits_{n=1,2,\cdots N_t} (\mathrm{Ham}(b_k^s, b_n^t)) \\ k = \arg \min\limits_{n=1,2,\cdots N_s} (\mathrm{Ham}(b_h^t, b_n^s)) \end{cases} \tag{5}$$

where $\mathrm{Ham}(b_k^s, b_n^t)$ is the hamming distance between b_k^s and b_n^t. That means not only b_h^t is the most similar descriptor of b_k^s, but also b_k^s is the most similar descriptor of b_h^t. After all descriptors in B_s are matched against these descriptors in B_t according to Eq. (5), we obtain a set of

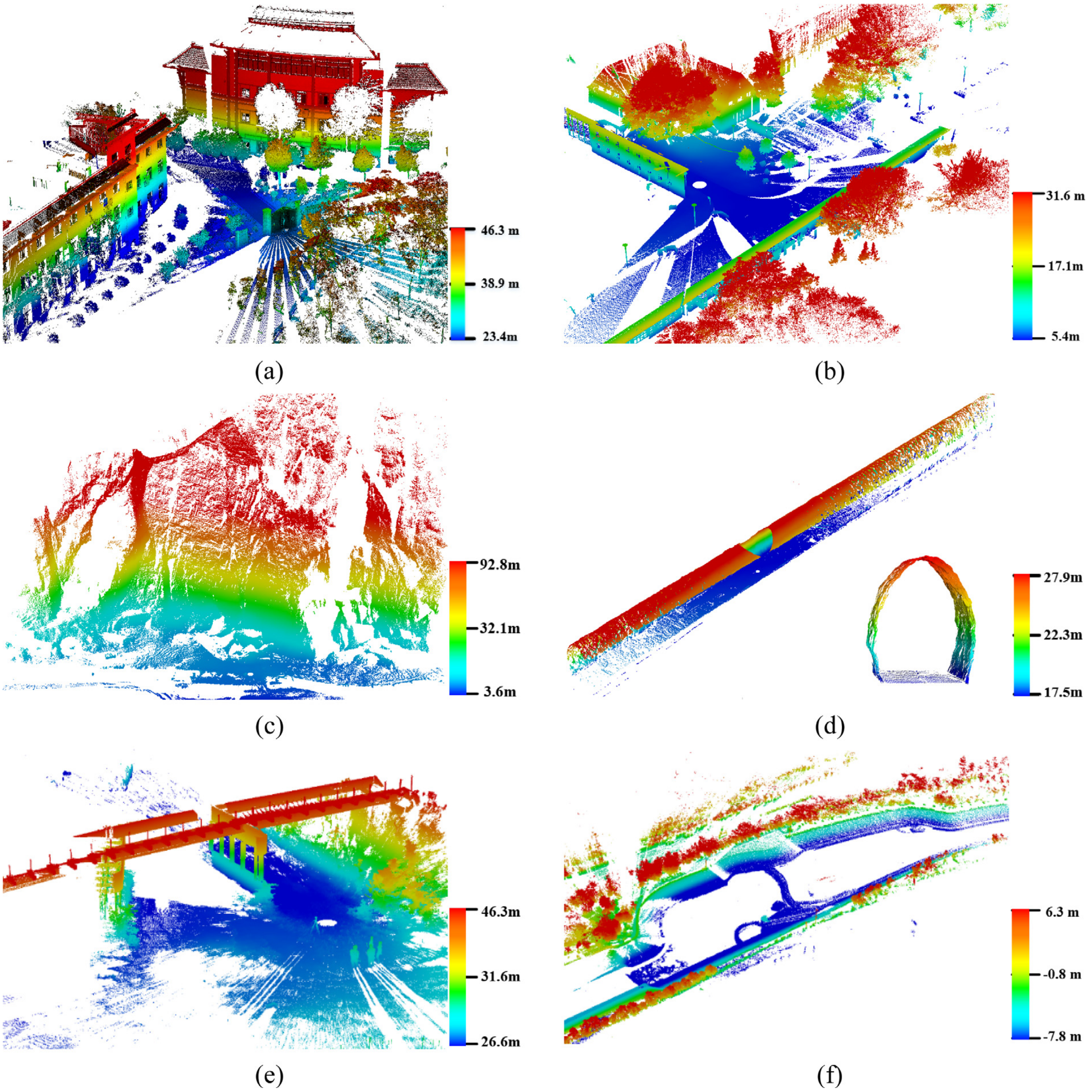


Fig. 6. The sampled point clouds from six datasets, with shading representing differences in height: (a) the 2nd point cloud of the park dataset, (b) the 98th point cloud of the campus dataset, (c) the 18th point cloud of the mountain dataset, (d) the 25th point cloud of the tunnel dataset and its cross section (one side of the tunnel was removed for a better visual effect), (e) the 1st point cloud of the bridge dataset and (f) the 3rd point cloud of the river dataset.

Table 2
Parameters of the proposed HMMR method.

Procedure	Parameter	Descriptor	Value
Multi-level descriptor calculation	*d*	The dimension of BSC descriptors	384
	K	The number of visual words	200
Overlapping point cloud pairs estimation	*q*	The number of the potentially overlapping point clouds for each point cloud	5
Hierarchical registration	*ε*	The threshold for geometric consistency verification	The average point distance
	Δ	The distance constraint for point cloud merge	The average point distance

In this paper, we calculate the distance from each point to its nearest point and regard the average of all the distances as the average point distance.

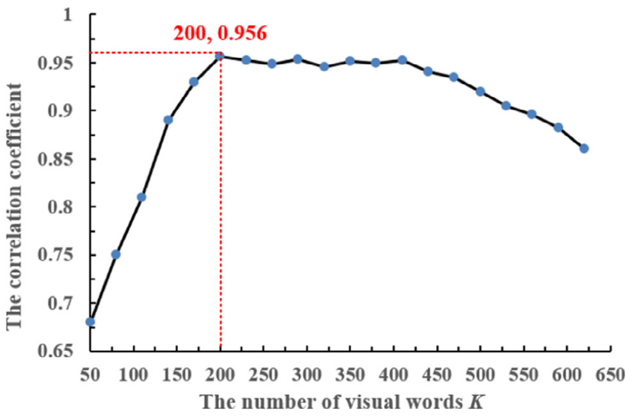


Fig. 7. The impact of *K* on the correlation coefficient between VLAD similarity and point clouds overlap.

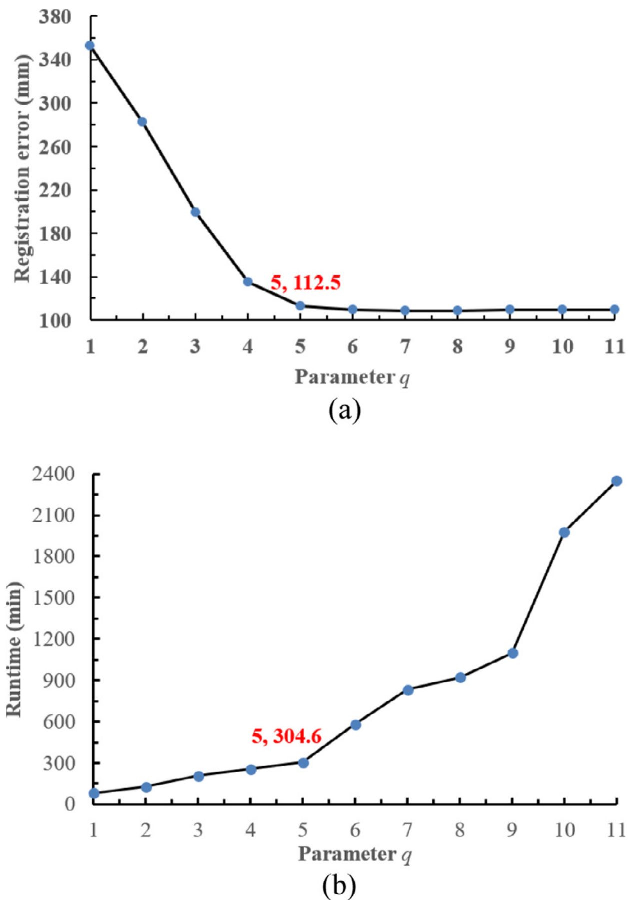


Fig. 8. The impact of parameter q on the registration error and runtime respectively.

correspondences $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$ between RP_s and RP_t, where $c_m\{b^s_{(m)}, b^t_{(m)}\}$ represent the mth correspondence, M_{FC} is the number of all the correspondences. Note that, the Hamming distance (i.e., XOR operation) is calculated extremely fast on modern CPUs, which makes the correspondence estimation process more efficient.

- **incorrect correspondences rejection**

Once the correspondence set $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$ between RP_s and RP_t is generated, we can estimate an optimal transformation according to FC. However, several incorrect correspondences may exist in $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$, which will subsequently result in wrong transformation estimations. Therefore, the geometric consistency verification (Chen and Bhanu, 2007) is used to further eliminate the incorrect correspondences in $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$. If two correspondence $c_m\{b^s_{(m)}, b^t_{(m)}\}$ and fulfill Eq. (6), they are considered as consistent correspondences. For a correspondence $c_m\{b^s_{(m)}, b^t_{(m)}\}$, the proposed method traverses all the other correspondences in $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$ and clusters those correspondences fulfilling Eq. (6) as a group. Repeat the same procedure for each correspondence in $FC\{c_1, \cdots c_m, \cdots c_{M_{FC}}\}$ to acquire M_{FC} groups. The larger the group is, the more likely it contains the true correspondences. Therefore the group with the largest correspondences is selected as the final correspondences, denoted by $GC\{c_1, \cdots c_m, \cdots c_{M_{GC}}\}$.

$$abs(\|p^s_{(m)} - p^s_{(n)}\| - \|p^t_{(m)} - p^t_{(n)}\|) < \varepsilon \tag{6}$$

where $p^s_{(m)}$, $p^t_{(m)}$, $p^s_{(n)}$, $p^t_{(n)}$ are the keypoint of $b^s_{(m)}$, $b^t_{(m)}$, $b^s_{(n)}$, $b^t_{(n)}$ respectively, $\|p^s_{(m)} - p^s_{(n)}\|$ is the Euclidean distance between $p^s_{(m)}$ and $p^s_{(n)}$, ε is a predefined threshold, abs represents the absolute value.

- **transformation calculation**

The correspondences $GC\{c_1, \cdots c_m, \cdots c_{M_{GC}}\}$ are used to calculate the pairwise coarse transformation $T^C_{s,t}$ between RP_s and RP_t by minimizing Eq. (7). Then, the fine transformation $T^F_{s,t}$ between them is further acquired by leveraging the pairwise level ICP algorithm (Besl and McKay, 1992) on the points belonging to RP_s and RP_t. Finally, the point cloud group PG_t and its BSC descriptors B_t are transformed to PG^*_t and B^*_t respectively based on the fine transformation $T^F_{s,t}$. Note that, the transformation of BSC descriptor only transforms its local coordinate reference with respect to $T^F_{s,t}$, the corresponding bit strings remains unchanged.

$$\delta = \sum_{l=1}^{L} \|p^s_{(l)} - T^C_{s,t} * p^t_{(l)}\| \tag{7}$$

where $T^C_{s,t}$ is the pairwise coarse transformation between RP_s and RP_t, $p^s_{(l)}$ and $p^t_{(l)}$ are the lth corresponding points in RP_s and RP_t respectively, L is

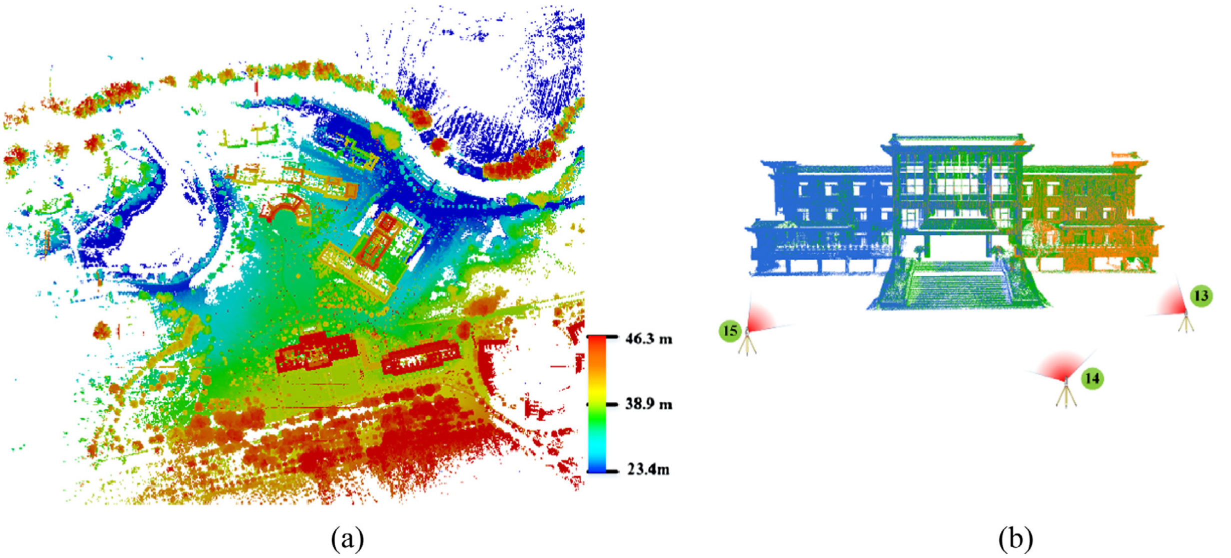


Fig. 9. Registration results of park dataset: (a) park dataset as aligned from all 32 point clouds, with shading representing differences in height (top view); (b) the details of the registration result in front view, where different colors stand for points from different point clouds.

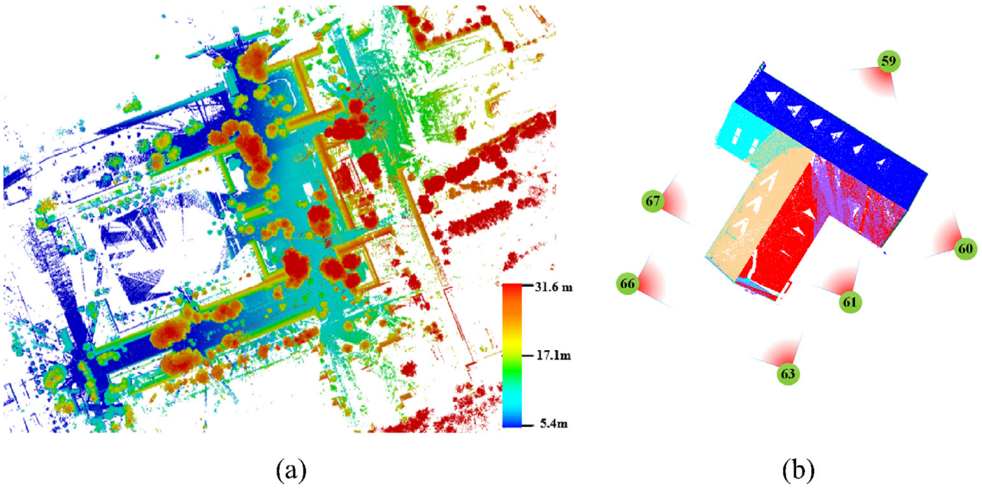


Fig. 10. Registration results of campus dataset: (a) campus dataset as aligned from all 132 point clouds, with shading representing differences in height (top view); (b) the details of the registration result in top view, where different colors stand for points from different point clouds.

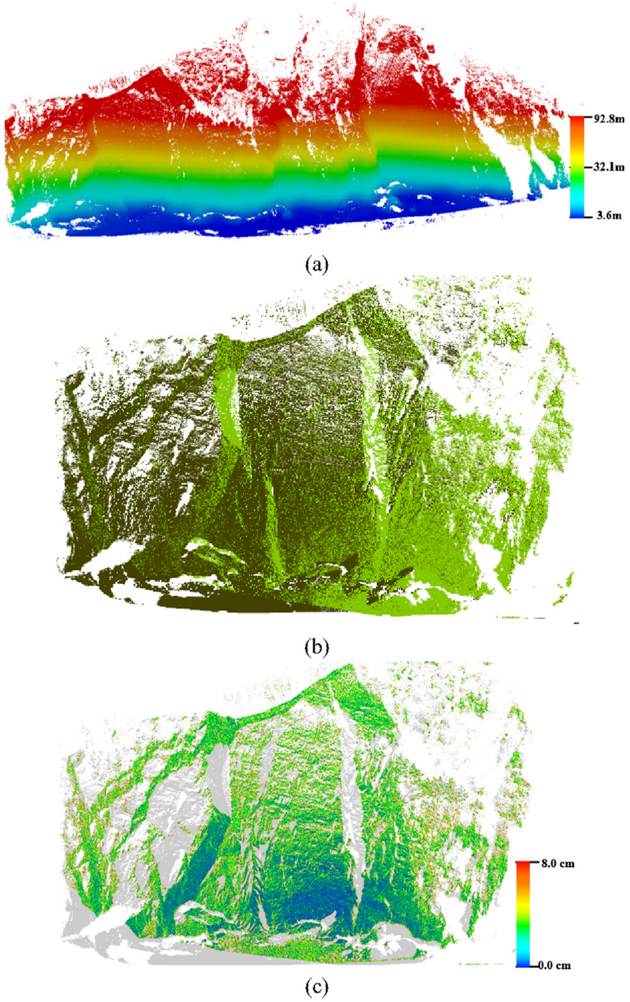


Fig. 11. Registration results of mountain dataset: (a) mountain dataset as aligned from all 22 point clouds, with shading representing differences in height (front view); (b) the details of the registration result in front view, where different colors stand for points from different point clouds; and (c) the details of the registration result colored by the residual distance between corresponding points, where the gray points represent the non-overlap areas.

the number of corresponding points in the largest group, and δ is calculated as the sum of the Euclidean distances between the corresponding points.

2.4.4. Point cloud groups overlapping (adjacent) graph update

After pairwise registration, we merge PG_s and PG_t^* into a new point cloud group $PG_{s,t}$ and calculate its BSC descriptors $B_{s,t}$ as follows: (1) all points of PG_s and all BSC descriptors of B_s are added to $PG_{s,t}$ and $B_{s,t}$ respectively. (2) The points of PG_t^* and the BSC descriptors of B_t^*, whose shortest distances to the points of PG_s are larger than a predefined threshold Δ, are added to $PG_{s,t}$ and $B_{s,t}$ respectively. The distance constraint is used to make sure that no redundant points and BSC descriptors are added to $PG_{s,t}$ and $B_{s,t}$. Note that, we generate BSC descriptors $B_{s,t}$ of $PG_{s,t}$ from the already available BSC descriptors of B_s and B_t^*, rather than recalculating brand-new descriptors. Therefore, this process improves the computational efficiency of BSC descriptors extraction as it does not need additional feature calculation during the process of hierarchical registration.

Once $PG_{s,t}$ and $B_{s,t}$ are acquired, the BSC descriptors $B_{s,t}$ and the K visual words (calculated at Section 2.2.2) are received as the input of Algorithm 1 to calculate the VLAD descriptor $V_{s,t}$ of $PG_{s,t}$. Then the similarity between $PG_{s,t}$ and its conterminal point cloud groups are recalculated according to Eq. (2). For example, in the first iteration, the source point cloud group PG_2 and the transformed target point cloud group PG_3^* are merged into a new point cloud group $PG_{2,3}$ and update the similarities between $PG_{2,3}$ and its conterminal point cloud groups (i.e., PG_1, PG_4, PG_5, PG_{10}), as shown in Fig. 5d. Repeat the process of Sections 2.4.2–2.4.4, until all the input point clouds are registered into a uniform coordinate reference. Finally, all the registered point clouds are further refined by using the multiview ICP algorithm (Williams and Bennamoun, 2001) on them to eliminate the error propagation.

Compared to the existing methods, the advantages of the hierarchical merging based algorithm are as follows. First, the optimal point clouds merging order is automatically determined according to the similarity between point clouds, resulting in the improvement of registration robustness and accuracy. Second, overlaps between multiple point clouds (as opposed to just pairwise overlaps) is leveraged during the point clouds registration process, which enhances the capability of handling point clouds with limited overlaps. For example, the multiple overlaps between point clouds (P_1, P_{12}^*), (P_2^*, P_{10}^*), (P_4^*, P_8^*) and (P_6^*, P_7^*) are leveraged to register the two point cloud groups in the last iteration, as shown in Fig. 5n.

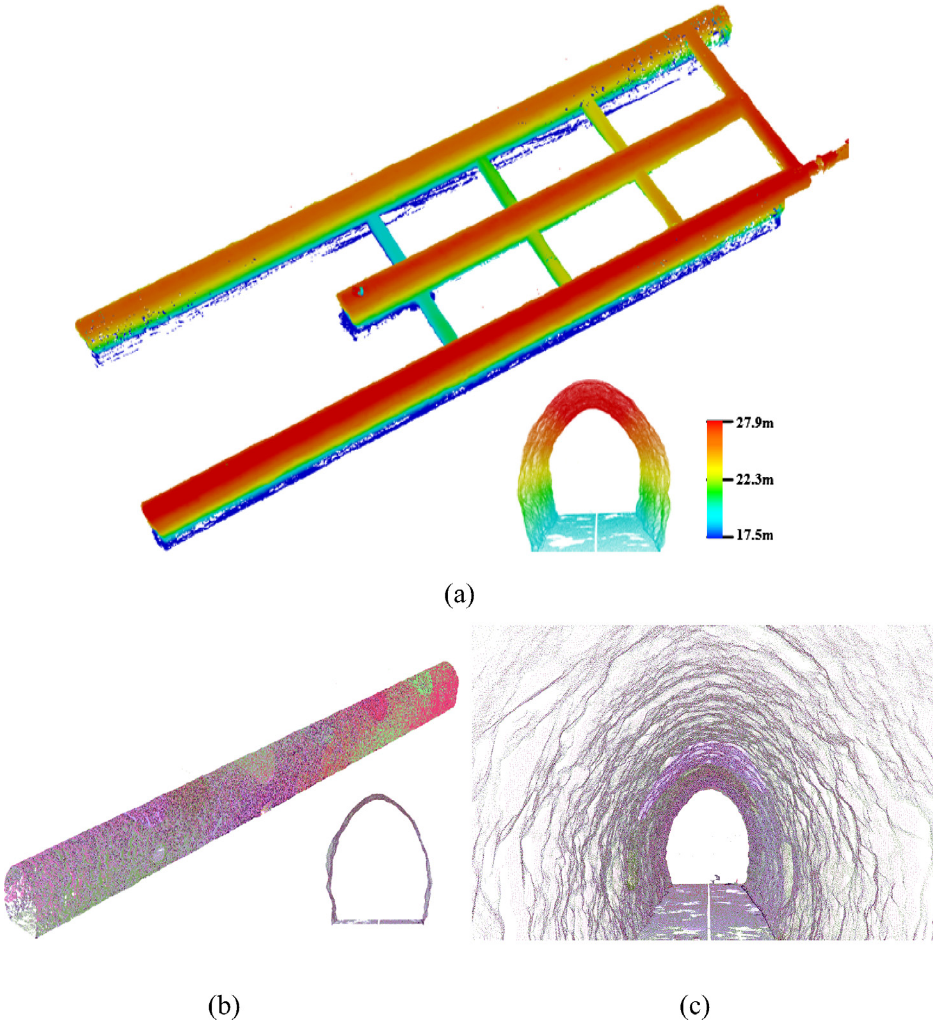


Fig. 12. Registration results of tunnel dataset: (a) tunnel dataset as aligned from all 26 point clouds, with shading representing differences in height (side view); (b) and (c) some details of the registration result in side view and front view of a tunnel, where different colors stand for points from different point clouds.

3. Experiments and analysis

3.1. Experimental setup

The implementation details of the experiments, including the description of datasets, the evaluation criteria, and parameter settings of the proposed method are described in this section.

3.1.1. Datasets description

The performance of the proposed HMMR algorithm is evaluated using six datasets derived from different scenes (i.e., park, campus, mountain, tunnel, bridge, and river). More specifically, the park, campus, tunnel, and river datasets were captured using the RIEGL VZ-400 laser scanner system with a field of view of 360° and 100° in the horizontal and vertical directions, a maximum range of 600 m, and a maximum measurement rate of 30 k pts/s. Mountain dataset was collected by Leica ScanStation C5 with all-in-one portability, a full 360° and 270° field-of-view, long range (300 m), and high scan speed (50 k pts/s). Bridge dataset was captured using a FARO FocusS 150 laser scanner with a field of view of 360° and 300° in the horizontal and vertical directions, a maximum range of 150 m, and a maximum measurement rate of 24 k pts/s. Table 1 provides a detailed description of the six datasets, and Fig. 6 shows the sampled point clouds from them. What makes these datasets challenging are (1) the datasets are collected by laser scanner systems with different field-of-view, measurement range, and accuracy; (2) each of the datasets contains billions of points; and (3) the datasets include a wide variety of scenes (e.g., park, mountain, river, tunnel) with significant disparity in land cover types and geometric structures of surface.

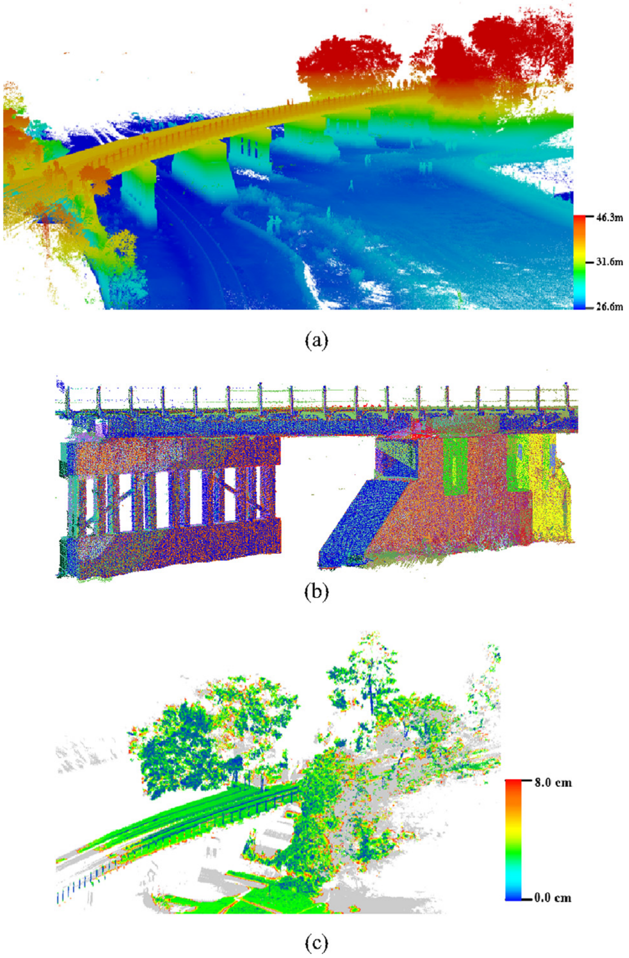
3.1.2. Evaluation criteria

We evaluate the performance of the proposed HMMR method in terms of rotation error, translation error, and successful registration rate which are commonly used for the evaluation of point cloud registration (Guo et al., 2014; Pomerleau et al., 2013; Theiler et al., 2014, 2015; Yang et al., 2016; Petricek and Svoboda, 2017). Given a reference point cloud P_r, the transformation $T_{r,i}$ from any point cloud P_i to P_r can be calculated by accumulating the corresponding pairwise transformations in HMMR. The residual transformation $\Delta T_{r,i}$ from any point cloud P_i to reference point cloud P_r is defined as Eq. (8):

$$\Delta T_{r,i} = T_{r,i}(T^G_{r,i})^{-1} = \begin{bmatrix} \Delta R_{r,i} & \Delta t_{r,i} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} \tag{8}$$

where $T_{r,i}$ are the estimated transformation from P_i to P_r, the $T^G_{r,i}$ is the corresponding ground-truth transformation.

Then the rotation error $e^r_{r,i}$ and translation error $e^t_{r,i}$ from P_i to P_r are calculated based on their corresponding rotational component $\Delta R_{r,i}$ and translational component $\Delta t_{r,i}$, as Eq. (9):



(a)

(b)

(c)

Fig. 13. Registration results of bridge dataset: (a) bridge dataset as aligned from all 29 point clouds, with shading representing differences in height (side view), and (b) a detail of the registration result with different colors standing for points from different point clouds (side view) and (c) the details of the registration result colored by the residual distance between corresponding points, where the gray points represent the non-overlap areas.

$$\begin{cases} e_{r,i}^{r} = \arccos\left(\frac{\mathrm{tr}(\Delta R_{r,i})-1}{2}\right) \\ e_{r,i}^{t} = \|\Delta t_{r,i}\| \end{cases} \tag{9}$$

where $\mathrm{tr}(\Delta R_{r,i})$ denotes the trace of $\Delta R_{r,i}$. The rotation error $e_{r,i}^{r}$ corresponds to the angle of rotation in the axis-angle representation.

Given the rotation error $e_{r,i}^{r}$ and translation error $e_{r,i}^{t}$, a successful registration (*SR*) is defined as Eq. (10):

$$SR = \begin{cases} 1 \ (e_{r,i}^{r} < \sigma_r) \wedge (e_{r,i}^{t} < \sigma_t) \\ 0 \ \text{else} \end{cases} \tag{10}$$

where σ_r and σ_t are the predefined threshold of rotation and translation error, which can be set according to the requirements of the applications. In this paper, the σ_r and σ_t are set as 100.0 millidegrees and 100.0 mm respectively. The successful registration rate (*SRR*) is then calculated as Eq. (11):

$$SRR = \frac{N_s}{N-1} \tag{11}$$

where N_s and N are the number of successful registration and the number of point clouds respectively.

3.1.3. Parameter settings

Table 2 shows the parameter settings of the proposed HMMR method, set by trial and error. All these parameter settings, unless otherwise specified, are used for all the experiments in this paper. The sensitivity of each parameter is tested on the campus dataset. The readers can refer to Section 3.1.1 for more details about the campus dataset.

There are five key parameters in the proposed registration algorithm, i.e., d, K, q, ε and Δ. The parameter d is the dimension of the BSC descriptor, which controls the descriptiveness and the efficiency of the BSC descriptor. Increasing the dimension of BSC descriptor can improve its descriptiveness to acquire more correct correspondences, but will decrease the computational efficiency and matching speed and increase the memory consumption. Therefore, we use $d = 384$ as the dimension of BSC descriptor to strike a balance between descriptiveness and efficiency. The readers can refer to Dong et al. (2017) for more details about the parameter settings of the BSC descriptors.

The parameter K is the number of visual words, which controls the descriptiveness and the compactness of the VLAD descriptor. We analyze the impact of the number K on the correlation coefficient between VLAD similarity and point clouds overlap, calculated as Eq. (12). Experimental results of the campus dataset find that the correlation coefficient dramatically increases as K increasing from 50 to 200, then remains relatively stable when K largens from 200 to 400, finally decreases slightly with the K ranging from 400 to 600, as shown in Fig. 7. Therefore, we use $K = 200$ as the number of visual words.

$$cc = \frac{Cov(O, S)}{Var(O)Var(S)} \tag{12}$$

where $Var(O)$ and $Var(S)$ are the variance of point clouds overlap and VLAD similarity respectively, $Cov(O, S)$ and cc are the covariance and correlation coefficient between VLAD similarity and point clouds overlap.

The parameter q is the number of the overlapping point clouds for each point cloud, which controls the efficiency and accuracy of the registration. More specifically, a small q may result in less accurate registration with less time, whereas a larger one may produce more accurate registration taking more time. Experimental results of the campus dataset find that the algorithm can strike a balance between registration error and runtime when q is set as 5, as shown in Fig. 8. Note that the registration error in Fig. 8a is calculated as the average distance between corresponding markers after registration.

The parameter ε is the threshold for geometric consistency verification, which is used to further eliminate incorrect correspondences. A small ε results in lots of omissive correspondences, whereas a large one produces some incorrect correspondences. The parameter Δ is the threshold for point cloud group merge, which makes sure that no redundant points are added to the merged point cloud group. Experimental results find that the proposed registration algorithm can obtain a good performance when ε and Δ are calculated as the average point distance from each point to its nearest point.

3.2. Experiment results

Figs. 9–14 show the outcomes and details of the proposed HMMR method on the six testing datasets respectively. Figs. 9–14a represent the global registration results on the six datasets, with shading representing differences in height. Other figures show some details about the registration results for selected regions. These qualitative experimental results show that the proposed HMMR method can acquire a good performance on all the six challenging datasets, demonstrating the feasibility of the proposed method for various scenes.

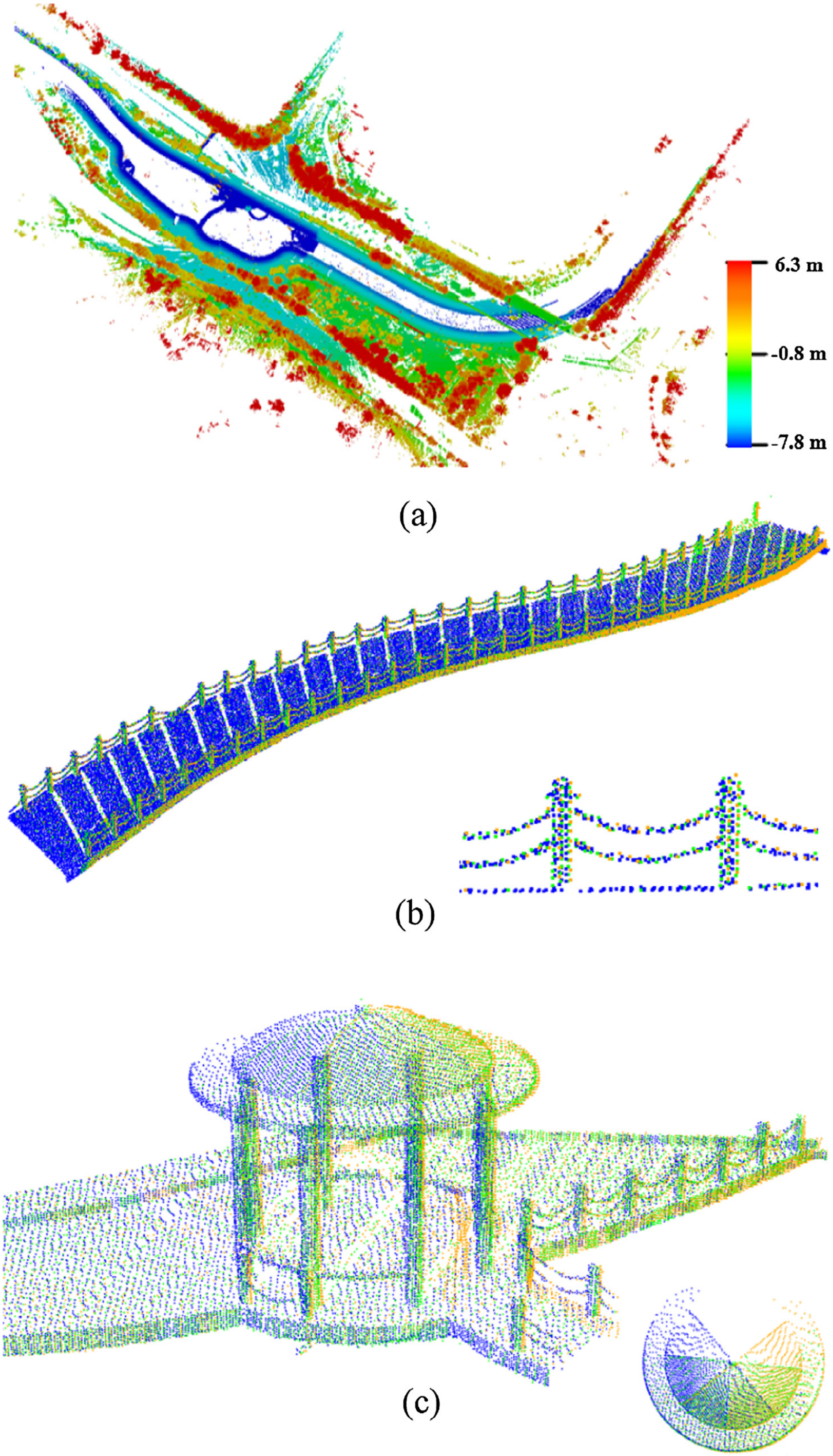


Fig. 14. Registration results of river dataset: (a) river dataset as aligned from all 13 point clouds, with shading representing differences in height (top view); (b) the registration result of a bridge on the river and a enlargement about the rail of the bridge, and (c) the registration result of a pavilion on the river and a top view about the roof of the pavilion, where different colors stand for points from different point clouds.

Table 3
Quantitative evaluation of the registration accuracy.

	Rotation error $e^r_{r,i}$ (mdeg)				Translation error $e^t_{r,i}$ (mm)				*SRR* (%)
	Min	Max	Ave	RMSE	Min	Max	Ave	RMSE	
Park	16.5	68.6	40.2	9.2	13.8	72.6	31.1	8.0	100.0
Campus	11.3	**142.4**	57.3	24.3	10.4	**210.6**	64.6	21.2	96.2
Mountain	13.2	63.1	36.7	8.7	25.1	72.9	33.8	11.4	100.0
Tunnel	6.7	57.3	37.2	9.8	10.4	**266.5**	60.2	19.8	92.0
Bridge	13.2	44.8	30.5	9.5	20.4	65.2	30.1	9.2	100.0
River	17.6	31.7	30.2	7.5	9.5	41.9	22.6	7.8	100.0

Table 4
Time performance of the proposed HMMR method.

Data	#Scans	#Point cloud pairs	# Exhaustive point cloud pairs	Runtime (min)			
				S1	S2	S3	Total
Park	32	69	$C^2_{32} = 496$	7.5	29.6	2.4	39.5
Campus	132	220	$C^2_{132} = 8646$	33.2	260.6	10.8	304.6
Mountain	22	51	$C^2_{22} = 231$	6.6	27.2	1.3	35.1
Tunnel	26	53	$C^2_{26} = 325$	8.7	34.5	2.4	45.6
Bridge	29	57	$C^2_{29} = 406$	9.6	37.9	2.7	50.2
River	13	25	$C^2_{13} = 78$	3.5	12.5	0.3	16.3

S1, S2, S3 indicate the processes of multi-level feature calculation, pairwise point cloud groups registration, and point cloud groups update respectively.

Table 5
The details of the benchmark methods.

Method	Local descriptor	Multiview registration strategy	Implementation
Weber et al. (2015)	FPFH	Minimum spanning tree	C++
MSTMR	BSC	Minimum spanning tree	C++
Guo et al. (2014)	RoPS	Shape growing	C++
SGMR	BSC	Shape growing	C++
HMMR	BSC	Hierarchical merging	C++

Table 6
Performance comparison.

Data		*SRR* (%)	Rotation error $e^r_{r,i}$ (mdeg)		Translation error $e^t_{r,i}$ (mm)		Runtime (min)
			Ave	RMSE	Ave	RMSE	
Park	Weber et al. (2015)	100.0	49.1	12.4	43.3	7.2	361.6
	MSTMR	100.0	48.6	9.8	42.9	8.1	177.8
	Guo et al. (2014)	100.0	44.5	13.7	37.2	9.4	186.2
	SGMR	100.0	44.7	13.6	36.8	9.8	94.6
	HMMR	100.0	**40.2**	9.2	**31.1**	8.0	**39.5**
Campus	Weber et al. (2015)	68.7	72.1	23.6	74.8	19.3	9214.6
	MSTMR	74.8	70.6	20.8	75.6	15.8	3869.5
	Guo et al. (2014)	75.6	65.3	19.6	67.7	16.2	3981.5
	SGMR	81.7	64.2	18.9	68.9	18.6	1839.6
	HMMR	**96.2**	**57.3**	24.3	**64.6**	21.2	**304.6**
Mountain	Weber et al. (2015)	85.7	60.2	16.4	52.8	19.3	322.6
	MSTMR	90.5	57.8	11.6	47.7	18.6	134.9
	Guo et al. (2014)	90.5	51.9	13.2	45.2	19.5	135.2
	SGMR	95.2	45.4	10.9	39.3	15.7	71.6
	HMMR	**100.0**	**36.7**	8.7	**33.8**	11.4	**35.1**
Tunnel	Weber et al. (2015)	60.0	67.4	20.8	75.4	29.1	413.6
	MSTMR	60.0	66.9	22.2	74.9	25.4	199.8
	Guo et al. (2014)	76.0	56.7	18.7	68.9	26.8	196.3
	SGMR	80.0	50.5	18.2	67.7	22.6	109.6
	HMMR	**92.0**	**38.5**	10.3	**61.3**	20.9	**45.6**

3.3. Evaluation and analysis

3.3.1. Registration accuracy evaluation

To evaluate the rotation and translation errors of the proposed HMMR method for TLS point clouds registration, the registration results are compared with those manually marked ground truths. To generate ground truth, point clouds are first aligned using manually marked point correspondences (Park dataset, Mountain dataset, and River dataset) or markers (Campus dataset, Tunnel dataset and Bridge dataset), then the manual registration is further refined by the multiview ICP algorithm (Williams and Bennamoun, 2001). The ground truth of Campus dataset is provided by Jacobs University.[1] Table 3 reports the rotation error and translation error (i.e., minimum, maximum, average, and the Root Mean Square Error (RMSE)) and the successful registration rate of the fine registration on the six testing datasets. Note that, only the successful registrations are utilized to calculate the average and RMSE of rotation and translation errors. These experimental results demonstrate that the proposed HMMR method performs well in registering the TLS point clouds from varying scenes, with average rotation error less than 0.1° and translation error less than 0.1 m, which can satisfy the requirements of object extraction and 3D reconstruction. More specifically, a number of observations can be noted based on these evaluation results. (1) The campus dataset obtained the worst performance with maximum rotation error 142.4 millidegrees and translation error 210.6 mm, which are greater than the predefined threshold of σ_r (i.e., 100.0 millidegrees) and σ_t (i.e., 100.0 mm). These larger registration errors are caused by repeatedly accumulating the small errors during the successive pairwise registration (Section 2.4). In the future work, we will further reduce these accumulated registration errors by an extension Lu-Milios algorithm (Borrmann et al., 2008), which optimizes the positions of all point clouds at once. (2) The tunnel dataset acquired the maximum translation error 266.5 mm (greater than the predefined threshold of σ_t), because the ICP algorithm is easily trapped into a local minimum due to the similarity along the tunnels. We will further figure out this problem by adding some constraints (i.e., the distance between corresponding keypoints) to the ICP algorithm.

3.3.2. Time performance analysis

A thorough efficiency evaluation of the proposed HMMR method is

[1] http://kos.informatik.uni-osnabrueck.de/3Dscans/.

also conducted in terms of the runtime in each step. The experiments are implemented on a computer with 16 GB RAM and an Intel Core i7-6700HQ @ 2.60 GHz CPU. Table 4 lists the number of point clouds, the number of the estimated and exhaustive point cloud pairs, and the runtime of each step. It is worth note that, the number of estimated point cloud pairs is notably smaller than the number of exhaustive point cloud pairs, thereby significantly improving the efficiency. It takes only about 5 h to automatically register the 132 point clouds collected during several weeks (i.e., the campus dataset with 2.75 billion points). The runtime shows that the proposed HMMR method acquires high efficiency in registering large-scale point clouds from various scenes, which will be detailedly analyzed in the next section.

3.3.3. Performance comparison and analysis

To further analyze the performance of the proposed HMMR method, its variants (i.e., **M**inimum **S**panning **T**ree-based **M**ultiview **R**egistration (MSTMR), **S**hape **G**rowing-based **M**ultiview **R**egistration (SGMR)) and the state-of-the-art registration methods (i.e., Guo et al., 2014, Weber et al., 2015) are selected as the benchmark methods for performance comparison. More specifically, MSTMR and SGMR respectively utilize the minimum spanning tree based strategy (Weber et al., 2015) and shape growing based strategy (Guo et al., 2014) for multiview registration instead of the proposed hierarchical merging strategy, all the other processes are consistent with the paradigm of HMMR. Table 5 lists the details (i.e., the local descriptors used for pairwise registration, the strategies used for multiview registration, and the programming language) of these benchmark methods for performance comparison. The parameters of all the benchmark methods are set according to the parameter settings recommended in the original articles Guo et al. (2014) and Weber et al. (2015). For the sake of consistent comparison, all the experiments are implemented in C + + and on a computer with 16 GB RAM and an Intel Core i7-6700HQ @ 2.60 GHz CPU. Table 6 lists their successful registration rate, registration errors and the runtime. It is found that the proposed method outperformed its variants and the state-of-the-art registration methods on the testing datasets. More specifically, a number of observations can be noted based on these comparison results.

(1) The hierarchical merging based multiview registration strategy achieved the best efficiency and was followed by shape growing based and minimum spanning tree based multiview registration strategies. The minimum spanning tree based methods (i.e., Weber et al., 2015, MSTMR) first performed exhaustive pairwise registration to construct a fully connected graph, then generated a minimum spanning tree of the graph to align all point clouds to the reference point cloud. For example, given the 132 point clouds in campus dataset, these methods need to repeat the pairwise registration process for $C_{132}^{2} = 8646$ times, which is very time-consuming and infeasible. The shape growing based methods (i.e., Guo et al., 2014, SGMR) first chose a seed point cloud and iteratively merged other point clouds with enough overlaps into the seed point cloud. Note that, the seed point cloud will become larger and larger during the iterations, hence registration of other point clouds with seed point cloud will become dramatically time-consuming. The high efficiency of our HMMR algorithm is attributable to the following three factors. First, the underlying point cloud pairs with larger overlaps are efficiently estimated by calculating the similarity between their corresponding VLAD vectors instead of exhaustive pairwise registration, which significantly reduces the computational complexity of registration. Second, the similarity between BSC descriptors can be measured by the Hamming distance (i.e., XOR operation), resulting in the BSC based pairwise registration faster than RoPS and FPFH based methods. Third, only the point clouds with larger overlaps (i.e., the calculated overlapping point cloud pairs in Section 2.3) instead of all the point clouds, are utilized during pairwise point cloud groups registration, which further improves the efficiency of registration.

(2) The hierarchical merging based multiview registration algorithm outperformed the other algorithms in terms of successful registration rate and registration error. The minimum spanning tree based methods (i.e., Weber et al., 2015, MSTMR) only leverage pairwise overlaps, therefore these methods have difficulty dealing with point clouds with limited pairwise overlaps. The shape growing based methods (i.e., Guo et al., 2014, SGMR) register target point cloud to seed point cloud by utilizing the one-to-many overlaps (seed point cloud contains multiple registered point clouds), which improves the performance in some extent. However, the shape growing based methods register target point cloud to seed point cloud according to the order of input point clouds instead of an optimal order, which may lead to larger registration error even incorrect registration. The good performance of our HMMR algorithm in terms of successful registration rate and registration error is attributable to the following three factors. First, the optimal point clouds merging/registering order is automatically determined according to the similarity between point clouds, resulting in the improvement of registration accuracy. Second, overlaps between multiple point clouds (as opposed to just pairwise overlaps) is leveraged during point clouds merging process, which enhances the capability of handling point clouds with limited overlaps. Third, MSTMR surpassed Weber et al. (2015) and SGMR outperformed Guo et al. (2014) in terms of successful registration rate and registration error, which indicates that the proposed BSC descriptor is more reliable than FPFH and RoPS for TLS point clouds registration.

4. Conclusions

At present, point cloud registration is at the core of many many applications, such as 3D model reconstruction, cultural heritage management, forest structure assessment, landslide monitoring, and solar energy analysis. This paper presented a hierarchical merging based multiview registration algorithm to align unordered point clouds from various scenes, and validated its performance on six challenging datasets. Comprehensive experiments demonstrated that the proposed hierarchical registration algorithm obtained good performances both in registration accuracy and runtime, and outperformed the state-of-the-art approaches. Although the proposed method can provide satisfactory registration results on the six challenging datasets, it still has difficulty in the registration of ambiguous scenes (e.g. interior corridor and forest). The interior corridor scene contains plenty of repetitive and symmetric structures, which may lead to some incorrect transformations with large overlaps. The forest scene includes a great deal of similar and unstable points (e.g., the points of the crown), which challenge the establishment of reliable points or feature correspondences. In future work, we will further modify the method to the interior corridor and forest scenes registration. The energy optimization strategy (Theiler et al., 2015), which first generates a set of putative transformations of pairwise registration then picks the best candidate transformations based on the global loop constraints, will be used to deal with the registration of interior corridor scenes with plenty of repetitive and symmetric structures. The possible solution for the forest scenes registration is to first remove the crowns then register the remaining point clouds. Other future works are as follows. First, the visibility consistency strategy proposed by Huber and Hebert (2003) can be utilized to verify the validity of each pairwise registration. Second, the extension Lu-Milios algorithm (Borrmann et al., 2008), which optimizes the positions of all point clouds at once, will be leveraged to reduce the accumulated registration errors. Third, the feasibility of the proposed algorithm for ALS point clouds and TLS point clouds registration will be investigated in our future work.

Acknowledgments

The authors would like to acknowledge Jacobs University for providing the campus dataset and the air lab of Carnegie Mellon University for providing the bridge dataset. This research is jointly supported by NSFC projects (Nos. 41725005, 41531177, 41501427), and National Key Technology Support Program (No. 2014BAL05B07).

References

Agarwal, S., Furukawa, Y., Snavely, N., et al., 2011. Building Rome in a day. Commun. ACM 54 (10), 105–112.

Aiger, D., Mitra, N.J., Cohen-Or, D., 2008. 4-Points congruent sets for robust pairwise surface registration. ACM Trans. Graph. 27 (3), 1–10.

Alahi, A., Ortiz, R., Vandergheynst, P., 2012. Freak: Fast retina keypoint. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 510–517.

Andres, B., Kappes, J., Beier, T., Kthe, U., Hamprecht, F., 2012. The lazy flipper: efficient depth-limited exhaustive search in discrete graphical models. In: European Conference on Computer Vision, pp. 154–166.

Asai, T., Kanbara, M., Yokoya, N., 2005. 3D modeling of outdoor environments by integrating omnidirectional range and color images. In: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05), Ottawa, ON, Canada, 13–16 June 2005, pp. 447–454.

Barnea, S., Filin, S., 2008. Keypoint based autonomous registration of terrestrial laser point-clouds. ISPRS J. Photogramm. Remote Sens. 63 (1), 19–35.

Bay, H., Ess, A., Tuytelaars, T., Gool, L., 2008a. Speeded-up robust features (SURF). Comput. Vis. Image Underst. 110 (3), 346–359.

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008b. Speeded-up robust features (SURF). Comput. Vis. Image Underst. 110 (3), 346–359.

Besl, Paul J., McKay, Neil D., 1992. Method for Registration of 3-D Shapes. International Society for Optics and Photonics, Robotics-DL Tentative, pp. 586–606.

Böhm, J., Becker, B., 2007. Automatic marker-free registration of terrestrial laser scans using reflectance. In: Proceedings of 8th Conference on Optical 3D Measurment Techniques, Zurich, Switzerland, July 9–12, 2007, pp. 338–344.

Borrmann, D., Elseberg, J., Lingemann, K., Nchter, A., Hertzberg, J., 2008. Globally consistent 3D mapping with scan matching. Robot. Auton. Syst. 56 (2), 130–142.

Chen, H., Bhanu, B., 2007. 3D free-form object recognition in range images using local surface patches. Pattern Recogn. Lett. 28 (10), 1252–1262.

Chen, M., Wang, S., Wang, M., Wan, Y., He, P., 2017. Entropy-based registration of point clouds using terrestrial laser scanning and smartphone GPS. Sensors 17 (1), 229–239.

Dold, C., Brenner, C., 2006. Registration of terrestrial laser scanning data using planar patches and image data. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 36 (5), 78–83.

Dong, Z., Yang, B., Liu, Y., Liang, F., Li, B., Zang, Y., 2017. A novel binary shape context for 3d local surface description. ISPRS J. Photogramm. Remote Sens. 130, 431–452.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Frahm, J.M., Fite-Georgel, P., Gallup, D., et al., 2010. Building Rome on a cloudless day. In: European Conference on Computer Vision. Springer, Berlin Heidelberg, pp. 368–381.

Frome, A., Huber, D., Kolluri, R., et al., 2004. Recognizing objects in range data using regional point descriptors. In: European Conference on Computer Vision. Springer, Berlin Heidelberg, pp. 224–237.

Furukawa, Y., Hernández, C., 2015. Multi-view stereo: a tutorial. Found. Trends® Comput. Graph. Vis. 9 (12), 1–148.

Ge, X., 2016. Non-rigid registration of 3D point clouds under isometric deformation. ISPRS J. Photogramm. Remote Sens. 121, 192–202.

Ge, X., 2017. Automatic markerless registration of point clouds with semantic-keypoint-based 4-points congruent sets. ISPRS J. Photogramm. Remote Sens. 130, 344–357.

Guo, Y., Sohel, F., Bennamoun, M., et al., 2013. Rotational projection statistics for 3D local surface description and object recognition. Int. J. Comput. Vis. 105 (1), 63–86.

Guo, Y., Sohel, F., Bennamoun, M., et al., 2015. A novel local surface feature for 3D object recognition under clutter and occlusion. Inf. Sci. 293 (2), 196–213.

Guo, Y., Sohel, F., Bennamoun, M., Wan, J., Lu, M., 2014. An accurate and robust range image registration algorithm for 3d object modeling. IEEE Trans. Multimedia 16 (5), 1377–1390.

Habib, A., Detchev, I., Bang, K., 2010. A comparative analysis of two approaches for multiple-surface registration of irregular point clouds. ISPRS – Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 38 (1), 61–66.

Huang, P., Cheng, M., Chen, Y., Zai, D., Wang, C., Li, J., 2017. Solar potential analysis method using terrestrial laser scanning point clouds. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10 (3), 1221–1233.

Huber, D.F., Hebert, M., 2003. Fully automatic registration of multiple 3D data sets. Image Vis. Comput. 21 (7), 637–650.

Jégou, H., Douze, M., Schmid, C., et al., 2010. Aggregating local descriptors into a compact image representation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 3304–3311.

Jegou, H., Perronnin, F., Douze, M., Sánchez, J., Perez, P., Schmid, C., 2012. Aggregating local image descriptors into compact codes. IEEE Trans. Pattern Anal. Mach. Intell. 34 (9), 1704–1716.

Johnson, A.E., Hebert, M., 1999. Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans. Pattern Anal. Mach. Intell. 21 (5), 433–449.

Jung, J., Hong, S., Jeong, S., et al., 2014. Productive modeling for development of as-built BIM of existing indoor structures. Autom. Constr. 2014 (42), 68–77.

Kelbe, D., Aardt, J.V., Romanczyk, P., Leeuwen, M.V., Cawse-Nicholson, K., 2016a. Marker-free registration of forest terrestrial laser scanner data pairs with embedded confidence metrics. IEEE Trans. Geosci. Remote Sens. 54 (7), 4314–4330.

Kelbe, D., van Aardt, J., Romanczyk, P., van Leeuwen, M., Cawse-Nicholson, K., 2016b. Marker-free registration of forest terrestrial laser scanner data pairs with embedded confidence metrics. IEEE Trans. Geosci. Remote Sens. 54 (7), 4314–4330.

Knopp, J., Prasad, M., Willems, G., Timofte, R., Van Gool, L., 2010. Hough transform and 3D SURF for robust three dimensional classification. In: Computer vision–ECCV 2010, pp. 589–602.

Kruskal, J.B., 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Am. Math. Soc. 7 (1), 48–50.

Leutenegger, S., Chli, M., Siegwart, R.Y., 2011. BRISK: Binary robust invariant scalable keypoints. In: International Conference on Computer Vision. IEEE, pp. 2548–2555.

Liang, X., Litkey, P., Hyyppa, J., Kaartinen, H., Vastaranta, M., Holopainen, M., 2012. Automatic stem mapping using single-scan terrestrial laser scanning. IEEE Trans. Geosci. Remote Sens. 50 (2), 661–670.

Lo, T.W.R., Siebert, J.P., 2009. Local feature extraction and matching on range images: 2.5 D SIFT. Comput. Vis. Image Underst. 113 (12), 1235–1250.

Mian, A., Bennamoun, M., Owens, R., 2006. Three dimensional model-based object recognition and segmentation in cluttered scenes. IEEE Trans. Pattern Anal. Mach. Intell. 28 (10), 1584–1601.

Mian, A., Bennamoun, M., Owens, R., 2010. On the repeatability and quality of keypoints for local feature based 3D object retrieval from cluttered scenes. Int. J. Comput. Vis. 89 (2), 348–361.

Montgomery, D.C., Peck, E.A., 2007. Introduction to linear regression analysis. J. R. Stat. Soc. 170 (3), 856–857.

Montuori, A., Luzi, G., Stramondo, S., Casula, G., Bignami, C., Bonali, E., Bianchi, M.G., Crosetto, M., 2014. Combined use of ground-based systems for Cultural Heritage conservation monitoring. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 4086–4089.

Nister, D., Stewenius, H., 2006. Scalable recognition with a vocabulary tree. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, pp. 2161–2168.

Oesau, S., Lafarge, F., Alliez, P., 2014. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. ISPRS J. Photogramm. Remote Sens. 2014 (90), 68–82.

Pajdla, T., Gool, L.V., 1995. Matching of 3-D curves using semi-differential invariants. In: Proceedings of the Fifth International Conference on Computer Vision, pp. 390–395.

Petricek, T., Svoboda, T., 2017. Point cloud registration from local feature correspondences—Evaluation on challenging datasets. PloS One 12 (11), e0187943.

Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S., 2013. Comparing ICP variants on real-world data sets. Autonom. Robots 34 (3), 133–148.

Prokop, A., Panholzer, H., 2009. Assessing the capability of terrestrial laser scanning for monitoring slow moving landslides. Natural Haz. Earth Syst. Sci. 9 (6), 1921–1928.

Pu, S., Li, J., Guo, S., 2014. Registration of terrestrial laser point clouds by fusing semantic features and GPS positions. Acta Geod. Cartogr. Sin. 2014 (43), 545–550.

Rabbani, T., Dijkman, S., van den Heuvel, F., et al., 2007. An integrated approach for modelling and global registration of point clouds. ISPRS J. Photogramm. Remote Sens. 61 (6), 355–370.

Rosten, E., Drummond, T., 2006. Machine learning for high-speed corner detection. In: European Conference on Computer Vision. Springer, Berlin Heidelberg, pp. 430–443.

Rusu, R.B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (FPFH) for 3d registration. In: IEEE International Conference on Robotics and Automation, ICRA'09. IEEE, pp. 3212–3217.

Sipiran, I., Bustos, B., 2011. Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes. Vis. Comput. 27 (11), 963–976.

Snavely, N., Seitz, S.M., Szeliski, R., 2008. Skeletal graphs for efficient structure from motion. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, pp. 2.

Stamos, I., Leordeanu, M., 2003. Automated feature-based range registration of urban scenes of large scale. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE II-555–II-561.

Theiler, P.W., Wegner, J.D., Schindler, K., 2015. Globally consistent registration of terrestrial laser scans via graph optimization. ISPRS J. Photogramm. Remote Sens. 2015 (109), 126–138.

Theiler, P.W., Wegner, J.D., Schindler, K., 2014. Keypoint-based 4-points congruent sets-automated marker-less registration of laser scans. ISPRS J. Photogramm. Remote Sens. 2014 (96), 149–163.

Theiler, P., Schindler, K., 2012. Automatic registration of terrestrial laser scanner point clouds using natural planar surfaces. ISPRS Ann Photogramm. Remote Sens. Spat. Inf. Sci 1, 173–178.

Tombari, F., Salti, S., Distefano, L., 2010. Unique signatures of histograms for local surface description. In: European Conference on Computer Vision. Springer, Berlin Heidelberg, pp. 356–369.

Vosselman, G., Maas, H.-G., 2010. Airborne and Terrestrial Laser Scanning. Whittles Publishing.

Weber, T., Hänsch, R., Hellwich, O., 2015. Automatic registration of unordered point clouds acquired by Kinect sensors using an overlap heuristic. ISPRS J. Photogramm. Remote Sens. 2015 (102), 96–109.

Weinmann, M., Weinmann, M., Hinz, S., Jutzi, B., 2011. Fast and automatic image-based registration of TLS data. ISPRS J. Photogramm. Remote Sens. 66 (6), S62–S70.

Williams, J., Bennamoun, M., 2001. Simultaneous registration of multiple corresponding point sets. Comput. Vis. Image Underst. 81 (1), 117–142.

Xu, Y., Boerner, R., Yao, W., Hoegner, L., Stilla, U., 2017a. Automated coarse registration of point clouds in 3d urban scenes using voxel based plane constraint. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 4.

Xu, Y., Tuttas, S., Hoegner, L., et al., 2017b. Geometric primitive extraction from point clouds of construction sites using VGS. IEEE Geosci. Remote Sens. Lett. 14 (3), 424–428.

Yang, B., Dong, Z., Liang, F., et al., 2016. Automatic registration of large-scale urban scene point clouds based on semantic feature points. ISPRS J. Photogramm. Remote Sens. 2016 (113), 43–58.

Yang, B., Zang, Y., 2014. Automated registration of dense terrestrial laser-scanning point clouds using curves. ISPRS J. Photogramm. Remote Sens. 2014 (95), 109–121.

Yang, J., Li, H., Jia, Y., 2013. Go-ICP: solving 3D registration efficiently and globally optimally. In: IEEE International Conference on Computer Vision, pp. 1457–1464.

Zai, D., Li, J., Guo, Y., Cheng, M., Huang, P., Cao, X., Wang, C., 2017. Pairwise registration of TLS point clouds using covariance descriptors and a non-cooperative game. ISPRS J. Photogramm. Remote Sens. 134, 15–29.

Zhong, Y., 2009. Intrinsic shape signatures: a shape descriptor for 3d object recognition. In: 12th International Conference on Computer Vision Workshops (ICCV Workshops). IEEE, pp. 689–696.