Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Novel 3D point set registration method based on regionalized Gaussian process map reconstruction*

Bo LI[1], Yu ZHANG[2], Wen-jie ZHAO[‡1], Ping LI[2]

[1]*School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, China*

[2]*State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China*

E-mail: jameslb20@hotmail.com; zhangyu80@zju.edu.cn; zhaowenjie8@zju.edu.cn; pli@iipc.zju.edu.cn

Received Aug. 31, 2019; Revision accepted Feb. 2, 2020; Crosschecked Mar. 20, 2020

Abstract: Point set registration has been a topic of significant research interest in the field of mobile intelligent unmanned systems. In this paper, we present a novel approach for a three-dimensional scan-to-map point set registration. Using Gaussian process (GP) regression, we propose a new type of map representation, based on a regionalized GP map reconstruction algorithm. We combine the predictions and the test locations derived from the GP as the predictive points. In our approach, the correspondence relationships between predictive point pairs are set up naturally, and a rigid transformation is calculated iteratively. The proposed method is implemented and tested on three standard point set datasets. Experimental results show that our method achieves stable performance with regard to accuracy and efficiency, on a par with two standard methods, the iterative closest point algorithm and the normal distribution transform. Our mapping method also provides a compact point-cloud-like map and exhibits low memory consumption.

Key words: Point set registration; Gaussian process; Intelligent unmanned system (IUS)
https://doi.org/10.1631/FITEE.1900457 **CLC number:** TP242.6

1 Introduction

In recent years, intelligent unmanned systems (IUSs) have been an area of rapidly increasing research interest. Mobile IUSs such as unmanned vehicles, aircraft, ground robots, and underwater explorers are being broadly used in several military and civilian domains, including surveillance operation, anti-terrorism mission, agriculture, and civil engineering inspection. To fulfill all of the complicated requirements for both outdoor and indoor environments, one of the many fundamental abilities with which an IUS should be equipped is that of autonomous navigation. This is because mobile IUSs rely extremely heavily on accurate knowledge of their positions in the environment for control, decision-making, and path-planning. Using combinations of a satellite-based global positioning system (GPS) and inertial sensors, many approaches have been developed for outdoor environments. However, GPS might be unreliable or even unavailable in most indoor environments, in addition to many parts of urban canyons or mountain areas. Relying exclusively on GPS for positioning might also bring safety problems owing to signal jamming, resulting in a potentially vulnerable system. A popular alternative solution is to consecutively estimate the ego-motion of the IUSs using data from onboard sensors. Specifically, when point cloud data are used, this approach is known as point set registration. The point cloud data represent a three-dimensional (3D) range scan

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 61673341, 61703366, and 11705026)
ORCID: Bo LI, https://orcid.org/0000-0001-6781-0743; Wenjie ZHAO, https://orcid.org/0000-0003-0760-2867
© Zhejiang University and Springer-Verlag GmbH Germany, part of Springer Nature 2020

expressed as a set of noisy spatial coordinates and are usually obtained from a laser range finder or from feature points extracted from visual images. Given two or more frames of point cloud data, point set registration seeks to find the rotation and translation that best align them.

Point set registration is an interdisciplinary research topic that intersects with other fields, such as clinical medicine, computer vision, and pattern recognition. There are distinctive differences in applying point set registration methods to IUS as compared to other fields. For example, when applying point set registration to clinical use, high accuracy is required, as it is life-critical; however, the requirement for efficiency is not as important as when it is being applied to IUS. The main goal of applying point set registration to pattern recognition is usually to rebuild a model of an object, where the point cloud data are homogeneous. Comparatively, point set registration as applied to IUS places more emphasis on estimating the location or trajectory of the IUS, and the point cloud data are relatively sparse and non-uniform. In mobile IUSs, point set registration methods are often combined with mapping or optimization techniques to form a simultaneous localization and mapping (SLAM) approach (Cadena et al., 2016). When a point set registration method is applied to mobile IUSs, the availability and convenience of embedding it into a complete SLAM pipeline should also be considered.

In view of the differences introduced above, classifications of point set registration methods as applied to mobile IUSs can be generated: scan-to-scan registration and scan-to-map registration (Hess et al., 2016). Scan-to-scan registration estimates the relative pose between two time-adjacent measurements based on two frames of point cloud data. Scan-to-map registration finds the relative pose between a current frame and the world frame and requires a consistent map. When the point cloud from the current frame is aligned into the world frame, it is fused with the world map as an updated frame. These two types of registration methods have respective advantages and disadvantages. The scan-to-scan registration methods can be fast but accumulate errors quickly, similar to an inertial measurement unit. As compared to scan-to-scan registration, scan-to-map registration can limit the accumulation of errors to a great extent but operates with slower speeds because it requires an extra process, i.e., map construction. These two types of registration methods also produce maps with different forms when they are combined into SLAM systems. When a scan-to-scan registration method is used in an SLAM approach, the low-precision poses obtained are rectified using back-end optimization methods (Grisetti et al., 2010), and the map is built by aligning the point cloud data to a unified world coordinate system. Thus, a map constructed by this type of approach is directly composed of raw point cloud data and has a point-cloud-like representation. In this study, we call this map representation a "point cloud map." The point cloud map is dense and visually intuitionistic. However, as the point cloud data must be stored to represent the map, the memory cost of the point cloud map is high. Owing to the high dimensionality of the map state vector, this type of map is also difficult to manipulate or update. In contrast, the mapping process associated with scan-to-map registration usually provides a map representation coinciding with the registration method. Most existing scan-to-map registration methods are based on a grid-cell-like map. This type of map representation divides the space into evenly spaced grids and models the probabilistic information of each grid cell. In this study, we call this type of map representation a "grid map." The grid map is flexible and requires much less memory for storage but does not express the map as explicitly as the point cloud map does. Using the grid map to represent the map is similar to covering the environment with a mosaic. The sharpness of the grid map depends on the size of the grid. A high-resolution grid map means high computational and memory cost. Thus, there is interest in developing a registration method that possesses the advantages of both scan-to-scan and scan-to-map methods.

In this paper, we propose a new type of 3D map representation. Based on this representation, we develop a new scan-to-map registration method. Using a regionalized Gaussian process (GP) map reconstruction method, we build a dense and point-cloud-like map by splitting a point cloud into sub-regions and modeling the points in each sub-region with a GP. We name this type of map representation a "GP map." Although the GP map is point-cloud-like (i.e., similar to the point cloud map), the dimensions of the map state vectors in each sub-region are fixed, making the map easy to store or update. Based

on this new type of map representation, the information of the map can be expressed as spatial coordinates. Therefore, the registration problem can be easily handled using singular value decomposition (SVD). Furthermore, the proposed registration method has faster computational speed than traditional scan-to-map approaches. The main contributions of this study are as follows:

1. A new type of map representation is proposed. The proposed map representation is dense (similar to the point cloud map), and its memory consumption is low.

2. A new 3D scan-to-map point set registration method is proposed based on the new type of map representation.

3. The robustness and efficiency of the proposed method are tested and compared against two benchmark algorithms on a series of challenging real-world datasets. The performance and mechanisms of the proposed method are thoroughly analyzed.

2 Related work

2.1 Existing scan-to-map registrations and map representations

As discussed in Section 1, most existing 3D map representations are based on grid maps. One of the most widely used two-dimensional (2D) grid map representations is the grid occupancy map (Thrun et al., 2005). The occupancy map represents a map as a field of random variables arranged in evenly spaced grid cells. OctoMap, proposed by Hornung et al. (2013), can be considered to be a 3D version of the 2D occupancy grid maps. It discretizes the space into voxels and uses octrees to model the probabilistic occupancy values in each voxel. Although this type of map is memory-efficient, few practical correlated registration methods have been developed. A normal distribution transform (NDT) map, also based on a voxel-type map, models the map information in each voxel using a normal distribution. The scan-to-map registration using 3D-NDT (Magnusson et al., 2007) calculates rigid transformation and rotation by solving a standard optimization problem and has appeared to be one of the most compelling registration methods for mobile IUSs. Stoyanov et al. (2012) improved the method by minimizing the distance between two 3D-NDT models instead of maximizing the likelihood of a set of points. Saarinen et al. (2013) proposed an NDT occupancy map based on the NDT map and developed a method for updating the NDT map.

2.2 Gaussian process for map construction

An inherent limitation of an NDT or grid map is that the space is divided into regular voxel cells. The discretization causes discontinuities at the edges of the cells, leading to uncertainty and imprecise registration results. One solution to this problem is to integrally model the environment using a GP. O'Callaghan and Ramos (2012) presented a continuous Bayesian occupancy map representation for 2D environments built on GPs, which can be taken as a continuous version of a traditional grid occupancy map. Kim and Kim (2013) applied local GPs to build a 3D continuous occupancy map. In their research, the data were divided into small-size clusters, and a GP was used for each cluster. To overcome the discontinuity problems at the boundaries of the local GPs, an overlapping cluster was proposed. Doherty et al. (2017) proposed an inference-based 3D occupancy mapping method. The proposed method was also combined with developments in sparse kernels and data structures for mapping. Guizilini and Ramos (2019) proposed a regression methodology for terrain modeling based on sparse projections to a reproducing kernel Hilbert space. In all of these works, the point coordinates $\boldsymbol{p} = (x, y, z)$ were associated with the occupancy values s, and the GP was used to represent the relationship $s = f(\boldsymbol{p})$ as a Gaussian distribution in the function space. The methods proposed in this study are essentially different from those in the aforementioned studies. We use coordinates from two dimensions as training points and the coordinates from the remaining dimension as the observations. Specifically, the functional relationship modeled here is $z = f(x, y)$, $y = f(x, z)$, or $x = f(y, z)$.

Several other works have used GPs to model environments with a functional relationship of $z = f(x, y)$. Vasudevan et al. (2009) applied a GP to model large terrains with laser range measurements. Plagemann et al. (2008) used a GP with a nonstationary covariance function for laser-based terrain mapping. Smith et al. (2010) produced an implicit, non-parametric representation for 3D laser-based mapping using GP regression. This can handle point

clouds that cannot be expressed with $z = f(x, y)$. Different from their work, our research goal concerns mapping the environments with more complex structures, i.e., the structures that cannot be modeled using a single GP. We propose a regionalized GP mapping algorithm by decomposing a space into sub-regions (cubes of the same size) and separately modeling the points contained in each sub-region with a GP.

2.3 Iterative closest point

The most well-known scan-to-scan point set registration method is the iterative closest point (ICP) algorithm proposed by Besl and McKay (1992). The ICP algorithm creates point correspondences between two point sets by finding the point pair with the closest distance and then calculates a rigid transformation between the correspondent points using the SVD method, which is analytical and easy to operate. The transformation is iteratively refined by repeatedly generating new correspondent point pairs until a certain error metric is smaller than a certain value. Although our method belongs to different types of registration methods, it has similarities to the ICP method, as the map constructed by our method is still a point cloud map. In our method, the rigid transformation is also calculated using SVD, as in the ICP method. Moreover, our registration approach proceeds in an iterative mode.

2.4 Our previous work

In our previous work (Li et al., 2020), based on a concept of regionalized GP map construction, we proposed a scan-to-map registration method for 2D situation and fed it into an SLAM framework. In this study, we extend it to a 3D situation. The 3D structure is more complicated, and the complexity of the GP grows cubically. To accommodate this method to a 3D situation, two major improvements are proposed. First, we use the difference between consecutive transformations as the error metric instead of the absolute difference between the eigenvalues of the covariance matrix of the correspondent points. Second, we use the "kd-tree" method proposed by Shen et al. (2006) to accelerate the process of the GP. The basic idea of this approach is that a pair of observations far away from each other are almost uncorrelated, and predictions at a test location can be performed using only a small number of neighboring points.

3 Approach

3.1 3D regionalized GP map construction

In this subsection, we introduce how the GP is used for the map construction process. The introduction starts with a brief review of GP regression (Rasmussen and Williams, 2006). Given a training dataset $\mathcal{D} = \{(u_i,\ \boldsymbol{w}_i)|i = 1,\ 2,\ \ldots,\ N\}$ of N pairs of locations $\boldsymbol{w}_i \in \mathbb{R}^d$ and noisy observations $u_i \in \mathbb{R}$, a regression obtains the predictive distribution for the realization at the test locations $\boldsymbol{w}_*$, as denoted by $f_* = f(\boldsymbol{w}_*)$. In GP regression, the noisy observations u_i are assumed to be given by $u_i = f(\boldsymbol{w}_i) + \varepsilon_i,\ i = 1, 2,\ \ldots,\ N$, where ε_i are independent, normally distributed noise terms. By defining $\boldsymbol{u} = [u_1, u_2, \ldots, u_N]$, $\boldsymbol{w} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N]$, and $\boldsymbol{w}_* = [\boldsymbol{w}_{1*}, \boldsymbol{w}_{2*}, \ldots, \boldsymbol{w}_{M*}]$, the joint distribution is given by

$$\begin{aligned} &P(f_*, \boldsymbol{u}) \\ &= \mathcal{N}\left(m(\boldsymbol{u}), \begin{bmatrix} \sigma^2\boldsymbol{I} + k\left(\boldsymbol{w}, \boldsymbol{w}\right) & k\left(\boldsymbol{w}, \boldsymbol{w}_*\right) \\ k\left(\boldsymbol{w}_*, \boldsymbol{w}\right) & k\left(\boldsymbol{w}_*, \boldsymbol{w}_*\right) \end{bmatrix}\right), \end{aligned} \tag{1}$$

where $k(\cdot, \cdot)$ is the covariance function, $m(\boldsymbol{u})$ is the mean value of $\boldsymbol{u}$, σ is the variance of ε, and $\boldsymbol{I}$ is the identity matrix. With the training points and selected test locations as inputs, the predictive mean at the test locations is given by Eq. (2) and their covariance matrix is given by Eq. (3):

$$\boldsymbol{u}_* = k\left(\boldsymbol{w}_*, \boldsymbol{w}\right)\left(\sigma^2\boldsymbol{I} + \boldsymbol{K}(\boldsymbol{w}, \boldsymbol{w})\right)^{-1}\boldsymbol{u}, \tag{2}$$

$$\begin{aligned} \boldsymbol{\sigma}_* =& k\left(\boldsymbol{w}_*, \boldsymbol{w}_*\right) - k(\boldsymbol{w}_*, \boldsymbol{w}) \\ &\cdot \left(\sigma^2\boldsymbol{I} + \boldsymbol{K}\left(\boldsymbol{w}, \boldsymbol{w}\right)\right)^{-1} k(\boldsymbol{w}, \boldsymbol{w}_*), \end{aligned} \tag{3}$$

where $\boldsymbol{K}(\boldsymbol{w}, \boldsymbol{w})$ is a matrix with elements $k(\cdot, \cdot)$.

In our previous work (2D situations), the u and w values are 2D coordinates of x or y, and the functional relationship is expressed as $y = f(x)$ or $x = f(y)$. In this study, we extend our theory to a 3D situation by taking the 3D spatial coordinates as u and w and changing the functional relationship to $z = f(x, y)$, $y = f(x, z)$, or $x = f(y, z)$. Thus, in a 3D situation, $\boldsymbol{w}_i \in \mathbb{R}^2$. As in 2D situations, the 3D functional relationship cannot be decided uniformly, as in other works. For example, a set of points measured from a plane perpendicular

to the x-y plane cannot be presented by $z = f(x, y)$. In this study, to deal with this issue, we employ a "divide and conquer" strategy: we divide the entire region into small sub-regions and explore the functional relationship in each sub-region separately. In Fig. 1, we use typical laser-based 3D point cloud data to illustrate our method. The entire region is restricted to a cuboid that includes all of the measurement points. Then, the region is decomposed into n abutting and evenly spaced cubes, as sub-regions Ω_i, $i = 1, 2, \ldots, n$. The side length of the cubes is l. We restrict the region to exactly cover all of the points, and the boundary is set as an integral multiple of l (Fig. 1a). After the division, the sub-regions that contain fewer than three points are removed. We use principal component analysis to find a normal direction for each subset of points. If $P_i = \{\boldsymbol{p}_{ij} = [x_{ij}, y_{ij}, z_{ij}]^{\mathrm{T}} | j = 1, 2, \ldots, m_i\}$ is the subset of the point cloud data that belong to Ω_i, the center of each subset of points is

$$\boldsymbol{g}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} \boldsymbol{p}_{ij}. \tag{4}$$

.

The covariance matrix $\boldsymbol{C}_i$ of the point positions is computed as

$$\boldsymbol{C}_i = \sum_{j=1}^{m_i} (\boldsymbol{p}_{ij} - \boldsymbol{g}_i)(\boldsymbol{p}_{ij} - \boldsymbol{g}_i)^{\mathrm{T}}, \tag{5}$$

and the eigenvalue decomposition of matrix $\boldsymbol{C}_i$ is

$$\boldsymbol{C}_i = [\boldsymbol{v}_{i1}\ \boldsymbol{v}_{i2}\ \boldsymbol{v}_{i3}] \begin{bmatrix} \lambda_{i1} & 0 & 0 \\ 0 & \lambda_{i2} & 0 \\ 0 & 0 & \lambda_{i3} \end{bmatrix} [\boldsymbol{v}_{i1}\ \boldsymbol{v}_{i2}\ \boldsymbol{v}_{i3}]^{\mathrm{T}}, \tag{6}$$

where λ_{ij} $(j = 1, 2, 3)$ are the eigenvalues of $\boldsymbol{C}_i$, and $\boldsymbol{v}_{ij}$ $(j = 1, 2, 3)$ are the corresponding normalized eigenvectors. We choose the eigenvector $\boldsymbol{v}_{i,\min}$ of $\boldsymbol{C}_i$ corresponding to the minimum eigenvalue $\lambda_{\min} = \min\{\lambda_{i1}, \lambda_{i2}, \lambda_{i3}\}$ as the normal direction, as the point cloud has the minimum variance along this direction. The functional relationship should be formulated as close to this direction as possible. To choose a proper relationship, the angles of the normal direction relative to the x, y, and z axes are calculated from

$$\left[\mathrm{angle}_{ix}\ \mathrm{angle}_{iy}\ \mathrm{angle}_{iz}\right]^{\mathrm{T}} = \arccos(\boldsymbol{v}_{i,\min}). \tag{7}$$

Then, the functional relationship between the coordinates in each sub-region is determined based on these angles. If $\min\{\mathrm{angle}_{ix}, \mathrm{angle}_{iy}, \mathrm{angle}_{iz}\} = \mathrm{angle}_{ix}$, the functional relationship is determined as $x_{ij} = f(y_{ij}, z_{ij})$, as is the case in Fig. 1b; the other two cases follow the same approach, and the functional relationship is determined as $y_{ij} = f(x_{ij}, z_{ij})$ (Fig. 1c) or $z_{ij} = f(x_{ij}, y_{ij})$ (Fig. 1d).

Once the functional relationships in each sub-region are determined, we calculate the predictions at the selected test locations along with the variances, using Eqs. (2) and (3). To maintain approximate continuity, the point predictions in each Ω_i are modeled around the mean value of u_i. In our approach, the predictions and their test locations are combined as predictive points. Each prediction is actually a linear combination of the observations in the sub-region. The coefficient of each element in the linear combination depends on the distance between the test location of the predictive point and the training locations. We use the data in Fig. 1 to illustrate the selection procedure for our map

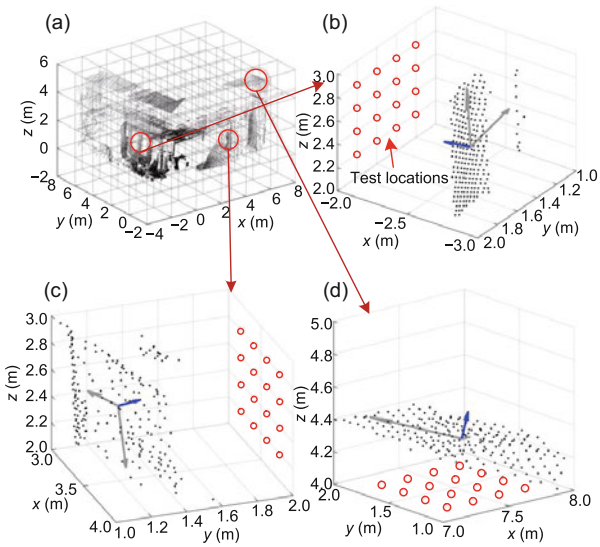

Fig. 1 Illustration of regionalization and the process for determining the functional relationship in sub-regions: (a) the entire raw point cloud; (b) a sub-region whose functional relationship is determined as $x = f(y, z)$; (c) a sub-region whose functional relationship is determined as $y = f(x, z)$; (d) a sub-region whose functional relationship is determined as $z = f(x, y)$. Black dots indicate the point cloud data, cubes visualized by gray thin lines indicate the boundaries of the sub-regions, small red circles indicate the selected test locations, big red circles indicate the exampled sub-regions with different determined functional relationships which are linked to corresponding detailed sub-figures with crimson arrows, blue arrows indicate the normal direction, and gray arrows indicate the other two eigenvectors. References to color refer to the online version of this figure

construction method (Fig. 2). Ambiguous predictions are always obtained at the test locations that are far away from the training locations of the observations (Fig. 2a). We ultimately obtain the regionalized GP predictive map by removing the predictions with variances larger than σ_{t} (Figs. 2b and 2c) and selecting the remaining predictive points as the results of the map construction procedure (Fig. 2d). Thus, in addition to parameter l, σ_{t} is another important parameter in our method.

Theoretically, each sub-region should possess an exclusive functional relationship if the size of the sub-regions is sufficiently small. However, the size cannot be set too small, as the computational speed will decrease excessively. Thus, the GP may generate some incorrect predictive points. They are generated in sub-regions containing point clouds measured from multi-surfaces or concave surfaces, i.e., situations for which our assumption of an exclusive functional relationship does not hold. However, these types of points make up only a very small portion of the entire set of predictive points (approximately 1%–5%).

3.2 Acceleration process based on kd-tree

Compared to our previous work for 2D situations, the number of points contained in one frame of the point cloud data herein is always much larger than that of a 2D point scan. As the complexity of the GP grows cubically, the computational cost could

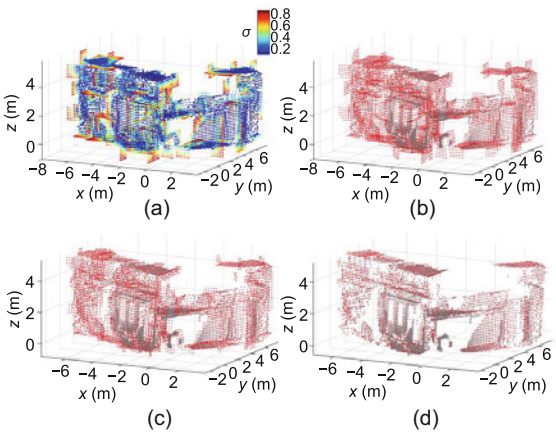

Fig. 2 Illustration of the prediction selection procedure: (a) scatter plots of all predictive points (colors indicate the value of σ); (b–d) predictive points for $\sigma_{\mathrm{t}} \leq 0.8$, $\sigma_{\mathrm{t}} \leq 0.5$, and $\sigma_{\mathrm{t}} \leq 0.1$, respectively. Red and black dots indicate the predictive points and the raw data, respectively. References to color refer to the online version of this figure

become too burdensome. To manage this problem, we employ the GP acceleration algorithm proposed by Shen et al. (2006). As discussed previously, the prediction at a test location is a linear combination of the observations in that sub-region. Taking this combination as a weighted sum, the prediction is contributed mostly by the observations with large weights, whose training locations are closer to the test location of that prediction. The central idea is to compute an approximation to that weighted sum while using only the observations with large weights. To that end, only the observations whose training locations are close to the selected test locations must be found. Hereafter, we denote these selected observations as "nearest points." As the process of searching for the nearest points is also time-consuming, a kd-tree data structure is employed to accelerate this process. Details can be found in Shen et al. (2006). In our case, kd-tree acceleration does not need to be performed for each sub-region. This is because a 3D point cloud is heterogeneous, especially one collected by a 3D laser range finder. We use only the kd-tree acceleration technique for sub-regions with large numbers of points. Assuming that the amount of point cloud data contained in a sub-region is m, the number of selected test locations is m_*, and the number of nearest points for one test location is k, kd-tree acceleration is performed only if $m \geq km_*$.

3.3 Registration based on GP predictive points

After the map construction process, the raw point cloud data are reformed as $\{(\boldsymbol{w}_{*,i}, u_{*,i}), \sigma_{*,i}\}$ $(i = 1, 2, \ldots, k)$, where $(\boldsymbol{w}_{*,i}, u_{*,i})$ are the special 3D coordinates composed of the selected test locations $\boldsymbol{w}_{*,i} \in \mathbb{R}^2$ and relative predictions $u_{*,i} \in \mathbb{R}$, and $\sigma_{*,i}$ are the variances. In this part, $(\boldsymbol{w}_{*,i}, u_{*,i})$ are regarded as predictive points and used for registration. As $(\boldsymbol{w}_{*,i}, u_{*,i})$ actually represents 3D points, the transformation between two frames of the point cloud can be calculated in an ICP-like manner. Specifically, the rigid transformation between corresponding predictive point pairs can be calculated using the SVD method.

The first task is finding the correspondences between predictive points. In contrast to the ICP, our registration method does not search for point correspondences by finding the nearest neighbor point with the minimum distance. In our approach, the

point correspondence is set up using the predictive point pairs with the same test locations and variances below the same threshold. For example, assume that there are two frames of unaligned raw point cloud data $\boldsymbol{Q}$ and $\boldsymbol{P}$, where $\boldsymbol{Q} = [q_1, q_2, \ldots, q_m]$ is the reference frame of the point cloud data, and $\boldsymbol{P} = [p_1, p_2, \ldots, p_n]$ is the current frame. Points $\boldsymbol{Q}$ and $\boldsymbol{P}$ are assumed to be placed in the same frame after an initial transformation. The initial transformation can be provided by other sensors, such as inertial measurement sensors, or by "coarse registration" (Salvi et al., 2007). Then, $\boldsymbol{Q}$ and $\boldsymbol{P}$ are reconstructed into GP predictive points $Q_* = \{\boldsymbol{q}_{i*} = (u_{i,q}, \boldsymbol{w}_{i,q}, \sigma_{i,q}) | i = 1, 2, \ldots, n\}$ and $P_* = \{\boldsymbol{p}_{i*} = (u_{i,p}, \boldsymbol{w}_{i,p}, \sigma_{i,p}) | i = 1, 2, \ldots, m\}$ using the map construction method described in Section 3.1. We construct the point correspondences by first identifying the overlapped sub-regions. Two sub-regions in a respective frame are considered to be overlapped only if they have the same cube center and the same functional relationship. When a pair of overlapped sub-regions are identified, the test locations for both sub-regions will be the same, according to the selection rule for test locations introduced in Section 3.1. The point correspondence in the sub-regions is set up using the predictive point pairs with the same test locations and variances below the same threshold σ_{t}. The process of building point correspondences in each sub-region is illustrated in Fig. 3. The final correspondent point pairs are an aggregation of all the correspondent point pairs in every overlapped sub-region.

By redefining P_* and Q_* in Cartesian coordinates as $P_* = \{([x_{i,p}, y_{i,p}, z_{i,p}]^{\mathrm{T}}, \sigma_{i,p}) | i = 1, 2, \ldots, k\}$ and $Q_* = \{([x_{i,q}, y_{i,q}, z_{i,q}]^{\mathrm{T}}, \sigma_{i,q}) | i = 1, 2, \ldots, k\}$, the corresponding point pairs can be expressed as $M(P_*, Q_*) = \{(\boldsymbol{p}_{i*}, \boldsymbol{q}_{i*}) | \boldsymbol{p}_{i*} \in P_*, \boldsymbol{q}_{i*} \in Q_*, i = 1, 2, \ldots, k\}$. As in the ICP method and our previous method, the rigid transformation is obtained by solving a Euclidean distance error minimization problem:

$$\arg\min_{\boldsymbol{R},\boldsymbol{t}} \left(\sum_{i=1}^{k} \|\boldsymbol{R}\boldsymbol{p}_{k*} + \boldsymbol{t} - \boldsymbol{q}_{k*}\|^2 \right). \tag{8}$$

This problem can be solved analytically using the SVD method. Assuming that $\boldsymbol{g}_p$ and $\boldsymbol{g}_q$ are the centers of $\{\boldsymbol{p}_{i*}\}$ and $\{\boldsymbol{q}_{i*}\}$, respectively, the covariance is

$$\boldsymbol{H} = \sum_{i=1}^{k} (\boldsymbol{p}_{i*} - \boldsymbol{g}_p)(\boldsymbol{q}_{i*} - \boldsymbol{g}_q)^{\mathrm{T}}. \tag{9}$$

Then the optimal transformations $\boldsymbol{R}$ and $\boldsymbol{t}$ can be computed by

$$\begin{cases} \boldsymbol{R} = \boldsymbol{V}\mathrm{diag}\left(1,1,\det\left(\boldsymbol{V}\boldsymbol{U}^{\mathrm{T}}\right)\right)\boldsymbol{U}^{\mathrm{T}}, \\ \boldsymbol{t} = \boldsymbol{g}_q - \boldsymbol{R}\boldsymbol{g}_p, \end{cases} \tag{10}$$

where $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}$ is the SVD of $\boldsymbol{H}$. Using Eq. (10), we obtain a rigid transformation composed of the rigid rotation matrix $\boldsymbol{R}$ and translation $\boldsymbol{t}$.

Similar to ICP approaches and many other registration methods, our registration approach proceeds in an iterative manner. When a transformation is calculated based on the corresponding predictive point pairs in a given iteration, it becomes the initial transformation for the next iteration, and the map construction process for the current frame of the point cloud data $\boldsymbol{P}$ is repeated to refine the transformation. The iterative procedure continues until an error metric is below a certain threshold.

As a termination condition, the error metric is crucial. In ICP-based approaches, the error metric is assigned as the sum of the squared distances between corresponding points or the distance from a point to the plane perpendicular to the destination normal (Chen and Medioni, 1991). As introduced in our previous work, these types of error metrics would not converge decreasingly as the iterative registration process proceeds in our approach, because we reconstruct the point data $\boldsymbol{P}$ using the GP map construction procedure in each iteration. In that regard, we propose a new error metric, called "E-error,"

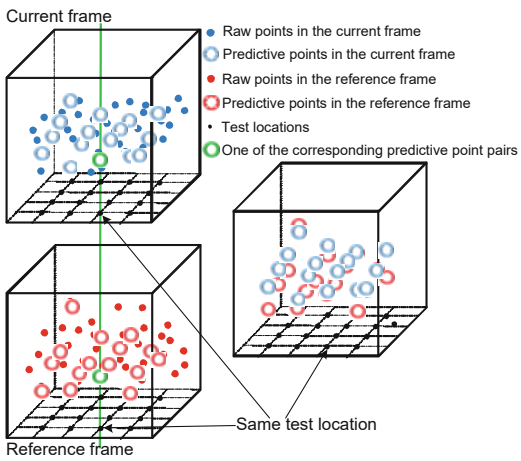

Fig. 3 Illustration of the point correspondence construction

which measures the difference between the eigenvalues of the covariance matrices of the corresponding predictive point pairs P_* and Q_*. The E-error works well for a 2D situation, but it appears to be unstable when moving to 3D. We propose a more intrinsic error metric, i.e., the difference between consecutive transformations. Assuming $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$ the transformations calculated in the current iteration, and $\boldsymbol{R}_{i-1}$ and $\boldsymbol{t}_{i-1}$ the transformations calculated in the last iteration, this error metric is given by

$$\begin{aligned}\text{error}\,(\boldsymbol{R},\boldsymbol{t}) &\triangleq \Delta T\\ &= \alpha\|\boldsymbol{R}_i-\boldsymbol{R}_{i-1}\|+\|\boldsymbol{t}_i-\boldsymbol{t}_{i-1}\|,\end{aligned} \tag{11}$$

where α is a scale factor.

In our approach, the termination condition for the iterative registration process is set as when error $(\boldsymbol{R},\boldsymbol{t})$ is smaller than a certain value, or when the iterative procedure has been executed more than 20 times.

4 Results

In this section, we demonstrate the results of the method described above. To fully and comprehensively test the performance of the proposed method, we chose a challenging dataset provided by Pomerleau et al. (2012). They claimed that the dataset is designed for testing the robustness of registration methods and covers various types of challenging environments, ranging from structured indoor environments to unstructured outdoor environments. Three sub-datasets were chosen for our research: "Stairs" (indoor, structured, many repeated elements) (Fig. 4a), "Gazebo" in summer (outdoor, semi-structured) (Fig. 4b), and "Wood" in summer (outdoor, unstructured) (Fig. 4c). For each sub-dataset, an adjacent point pair was selected, as shown in Figs. 4d–4f. The proposed method (GP registration) was compared to two benchmark algorithms, ICP and NDT. In the implementations of ICP and NDT the MATLAB functions provided by the MATLAB/Vision toolbox were used. In addition, OctoMap was used for benchmarking the mapping results. All methods were tested on a standard PC with a 3.4-GHz processor and run in a MATLAB-2019a environment. There were many parameters and factors for the three methods. To set up a uniform "baseline," we selected the point pairs

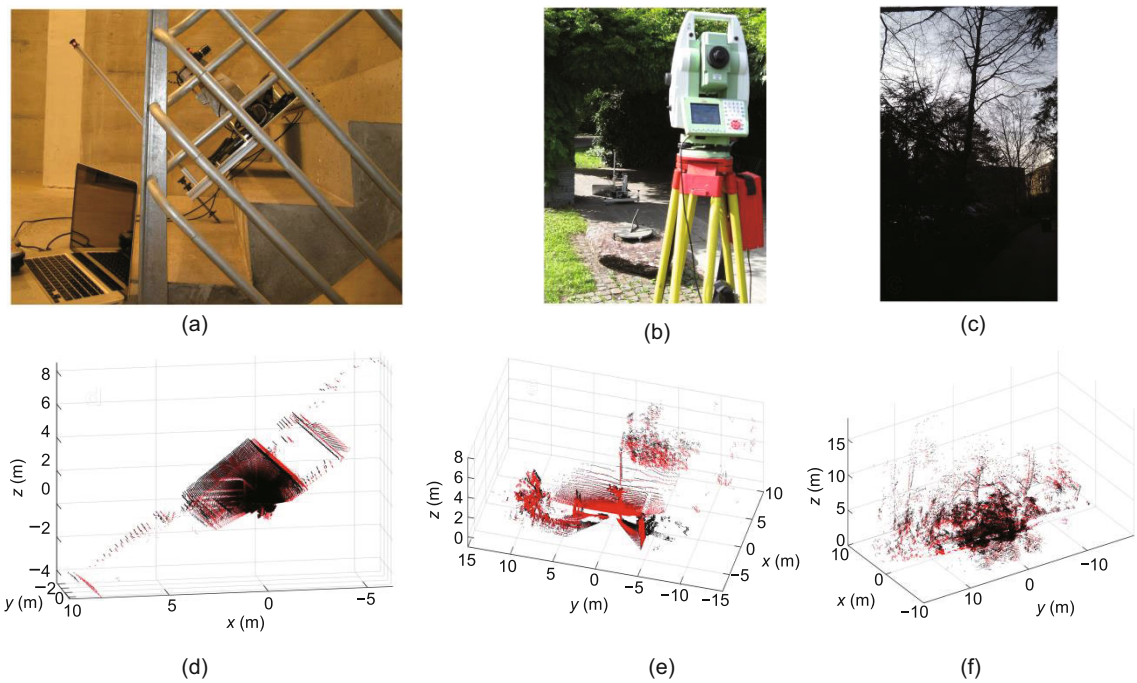

Fig. 4 Real-world views of the environments of the Stairs (a), Gazebo (b), and Wood (c) datasets, and point pairs from the Stairs (d), Gazebo (e), and Wood (f) datasets. (a)–(c) are directly downloaded from https://projects.asl.ethz.ch/datasets/doku.php?id=laserregistration:laserregistration. The black and red dots in (d)–(f) indicate the point cloud pair selected from each sub-dataset. References to color refer to the online version of this figure

with the minimum relative distance, so that we can set the initial transformation as the identify matrix $\boldsymbol{I}$ for all of the methods. In addition, there was not any down-sampling process for any of the methods during implementation.

4.1 Mapping results

Here, we use a frame of point cloud data from Gazebo as an example to illustrate our mapping results. Three other types of map representation are shown for comparison: raw point cloud, NDT map, and OctoMap. Different information is stored for each map representation. The raw point cloud stores all of the 3D coordinates of the point cloud. The NDT map stores the statistical information in each NDT cell, including the mean value (center) of the points contained in one cell and the covariance matrix of these points. OctoMap stores the occupancy value in each OctoMap cell. Our GP map stores the coordinates of the predictive points and their variances. The GP map (Fig. 5b) and the raw point cloud (Fig. 5a) are directly visualized as points. The NDT map is visualized by a set of ellipses contained in each cell, whose center is the mean value of the points, and whose three axes are the eigenvectors of the covariance matrix of the points (Fig. 5c). OctoMap is shown as a set of cubes representing the occupied cells (Fig. 5d). As shown in the figure, the raw point cloud provides all information from the measurement, and is the most detailed representation. The other three types of map representation are all certain types of deformation of the raw point cloud. Very similar to the point cloud map, the GP map appears to be a down-sampled variant of the raw point cloud map, as there are over 170 000 points in the point cloud map, whereas there are 14 767 predictive points in the GP map. However, the GP map inherits all of the structural signature, without losing details. The NDT map and OctoMap do not express the map as explicitly as the point cloud map and the GP map do. They visualize the map in a manner similar to covering the environment with a mosaic.

The respective resolutions of the NDT map and OctoMap are equal to the length of their cube cells. This is different from the GP map, as the predictive points are used for the final mapping result in the GP

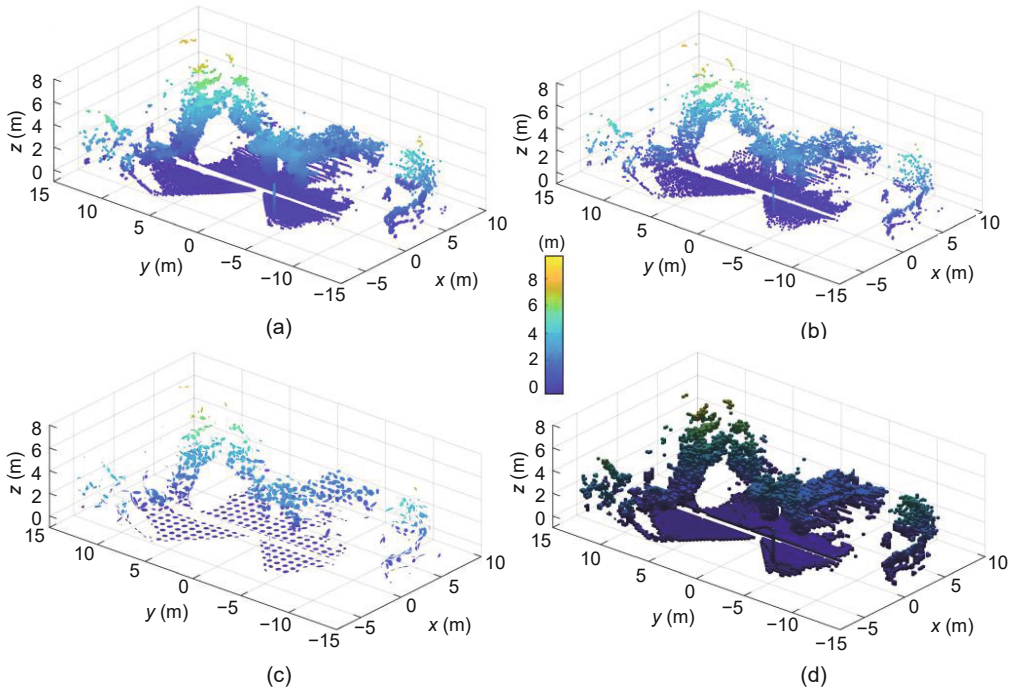

Fig. 5 Mapping results of the point cloud map (a), GP map (b), NDT map (c), and OctoMap (d). Colors indicate the height. The normal distribution in each cell including the mean value of the points and the covariance matrix of the points is visualized as ellipses. References to color refer to the online version of this figure

map. Thus, the resolution of the GP map depends on the density of the test locations. In the GP map, assuming that the side length of a GP cell l is one and that it is divided evenly into 10 segments, the square where the test location is located will be cut into 10×10 pieces, and the resolution will be 0.1. In Fig. 5, the resolution of the NDT map is set as 0.8 m, whereas the resolutions of the GP map and OctoMap are both 0.2 m. We set these values because the ellipses of the NDT map would be too small to be shown clearly if the resolution was high.

By clarifying the definition of resolution, the storage consumption of each method can be compared (Table 1). As mentioned before, the point cloud map stores the coordinates of the points and requires the maximum amount of memory. Though point-cloud-like, the GP map lowers the memory consumption dramatically. It requires only 401 KB of memory to store a point-cloud-like map with a resolution of 0.2 m. The NDT map also needs much less memory than the point cloud map, but it needs more memory than the GP map and is almost twice as large at resolutions of 0.2, 0.5, and 1.0 m. With the help of the efficient octree data structure, OctoMap consumes the minimum amount of memory. The memory cost of OctoMap is an order of magnitude smaller than that of the GP map and the NDT map. However, it has no matched efficient registration method. As compared to the NDT map and OctoMap, which are well-known for consuming less memory, our mapping method shares the same advantage while providing a point-cloud-like map.

Table 1 Memory consumption for each map representation with different resolutions

Resolution	Memory			
(m)	GP	NDT	OctoMap	Point cloud
0.1	1.38 MB	1.42 MB	129 KB	3.2 MB
0.2	401 KB	787 KB	29.6 KB	
0.5	93.5 KB	203 KB	5.35 KB	
1.0	45.2 KB	68.5 KB	2.21 KB	

4.2 Convergence results

As all the three registration methods calculate the transformations iteratively, convergence performance is an important criterion for evaluating the robustness of each method. Convergence refers to the process in which the error metric converges to a certain value as the iteration proceeds, and its performance is affected by the parameters. As introduced before, the two most important parameters for our method are the length of the sub-region l and the threshold of variance σ_{t}. For the NDT method, the parameter with the largest effect is the NDT cell size l_{n}. Notably, both our method and NDT divide the space into small cubes; l and l_{n} are actually the same entity here, but they have a different significance for each method. The situation of the ICP method is rather simple. With the preset conditions introduced at the beginning of this section, there is no special parameter that requires adjustment.

The convergence performances with different datasets are shown in Figs. 6 (Stairs), 7 (Gazebo), and 8 (Wood). As the convergence situation changes with the parameter when implementing the NDT and GP registrations, multi-curves are shown to illustrate the convergence results with different selected parameters. Figs. 6c, 7c, and 8c show the results of GP registration, in which parameters σ_{t} and l are selected as 0.02–0.10 and 0.3–1.0 m, respectively. As can be seen from these figures, the ICP approach demonstrates the most stable convergence performance among all three types of methods. Our GP registration method converges decreasingly at most parts of the convergence curve, but the convergence rate slows as the complexity of the environments grows. In the Stairs and Wood experiments, the convergence curves show some increases at the beginning of the iterative process. This phenomenon indicates that the point correspondence between the predictive point pairs may have some significant changes in the first few iterations of the iterative process. As compared to the ICP method and our method, the convergence curves of NDT in all three experiments are rather irregular, showing some instability in the registration process. The difference between sequential transformations estimated by NDT often changes drastically at the beginning and suddenly converges close to zero. Moreover, the difference between the transformations of two consecutive iterations of NDT can be zero. This means that the convergence process would become stuck in a local minimum, requiring the termination of the iterative registration process. This phenomenon never occurs during the implementation of ICP or GP registration. Thus, NDT registration often stops before the number of iterations reaches the maximum value.

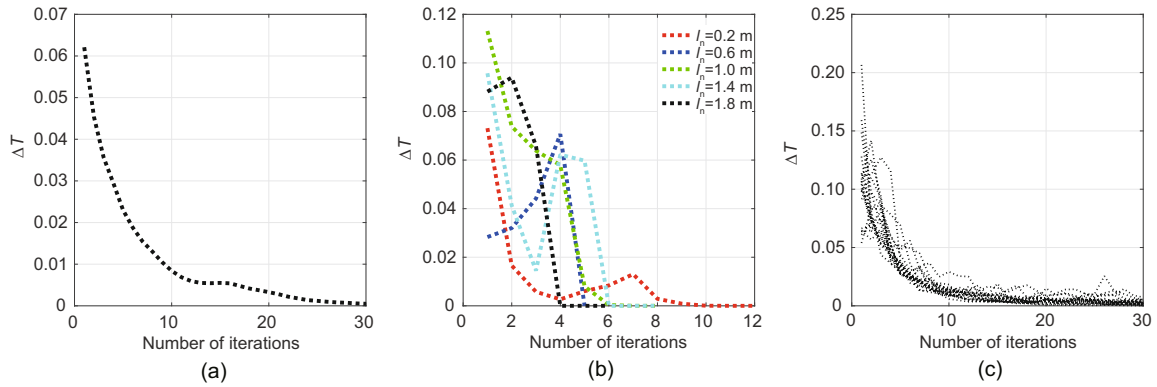

Fig. 6 Convergence results of experiment Stairs: (a) ICP; (b) NDT; (c) GP registration with different σ_t and l

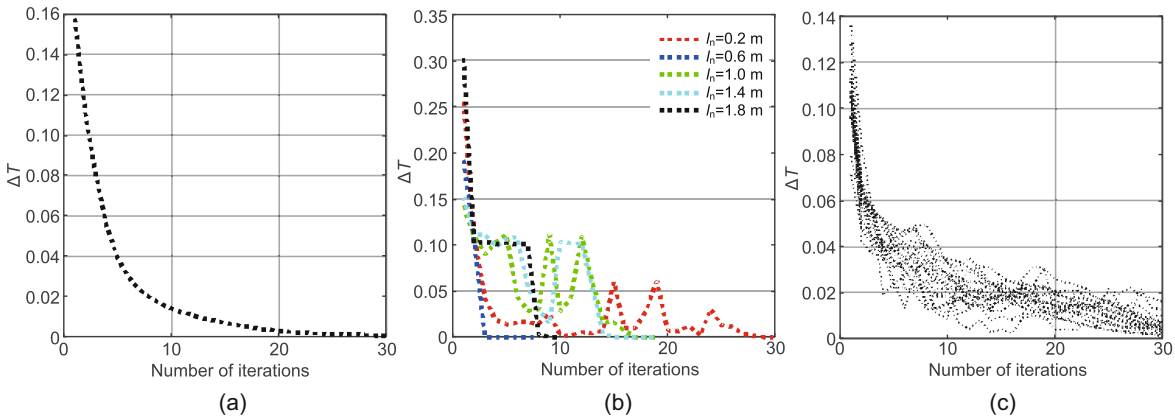

Fig. 7 Convergence results of experiment Gazebo: (a) ICP; (b) NDT; (c) GP registration with different σ_t and l

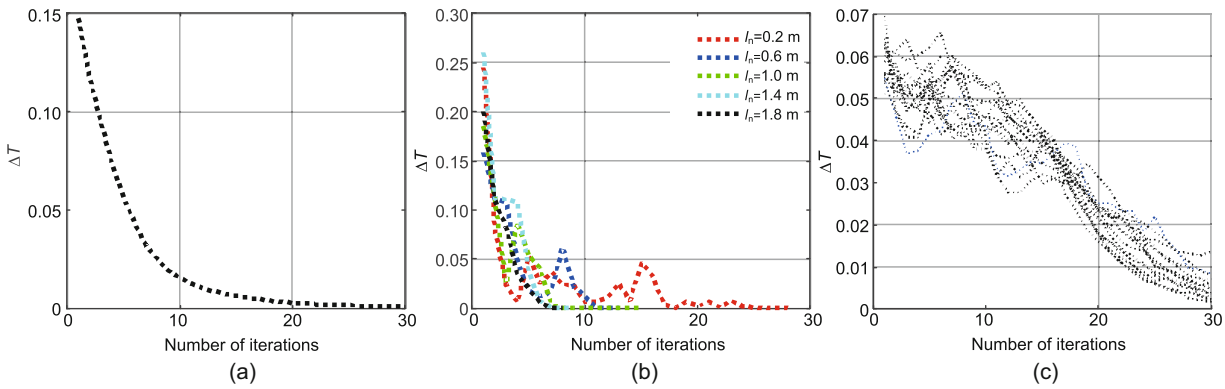

Fig. 8 Convergence results of experiment Wood: (a) ICP; (b) NDT; (c) GP registration with different σ_t and l

Though the NDT method has a faster convergence speed, this does not mean that NDT has a faster computational speed. A comparison between the efficiencies of the three methods will be further discussed in Section 4.3. By analyzing the convergence performance, we can set the termination conditions for each method as follows: for the ICP method and our method, parameter α in Eq. (11) is set at 5, and

the threshold of the error metric ΔT is set to be 0.001; for NDT, α is set at 0.2, and the threshold of ΔT is set at 0.0001, based on the value of α.

4.3 Registration results

Our registration method was tested using the three aforementioned datasets, with different parameters selected. By selecting σ_{t} and l from small to large, the performance of the proposed method was thoroughly revealed. All of the results of GP registration from the three experiments, including the errors in rotation (unit: rad), errors in translation (unit: m), and runtime (unit: s), are listed in Tables 2 (Stairs), 3 (Gazebo), and 4 (Wood). From these results, three main conclusions of regularity in our method can be summarized as follows:

1. Registration results with large errors in both rotation and translation might be obtained when l and σ_{t} are set to be too small or too large.

Because we use the predictive points to build the GP map and to calculate the rigid transformation, setting σ_{t} small means that only a small portion of the predictive points will be used for the calculation. With most of the predictive points abandoned, the information depicting the spatial relationships between point clouds is inadequate, leading to transformation results with large errors. As mentioned before, our method models the GP of each sub-region around the mean value $\boldsymbol{u}_i$ of each sub-region to approximately maintain the continuity of the GP map. Setting l to be too small would cause the discontinuity problem to become severe. Owing to the

Table 2 Results of GP registration in the Stairs experiment

σ_{t}	Rotation error (rad)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	0.0140	0.0110	0.0096	0.0057	0.0058	0.0012	0.0209	0.0222
0.03	0.0080	0.0011	0.0062	0.0041	0.0020	0.0005	0.0199	0.0202
0.04	0.0130	0.0010	0.0053	0.0024	0.0022	0.0006	0.0197	0.0197
0.05	0.0082	0.0009	0.0022	0.0019	0.0027	0.0013	0.0199	0.0190
0.06	0.0017	0.0014	0.0018	0.0018	0.0028	0.0014	0.0202	0.0110
0.07	0.0019	0.0017	0.0013	0.0014	0.0035	0.0017	0.0203	0.0036
0.08	0.0016	0.0021	0.0019	0.0013	0.0035	0.0019	0.0184	0.0029
0.09	0.0018	0.0023	0.0020	0.0014	0.0035	0.0021	0.0085	0.0028
0.10	0.0013	0.0028	0.0024	0.0012	0.0034	0.0021	0.0071	0.0028
σ_{t}	Translation error (m)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	0.0280	0.0035	0.0072	0.0027	0.0108	0.0065	0.0117	0.0247
0.03	0.1550	0.0042	0.0079	0.0048	0.0041	0.0034	0.0150	0.0187
0.04	0.0390	0.0027	0.0104	0.0054	0.0067	0.0047	0.0169	0.0119
0.05	0.0158	0.0040	0.0040	0.0059	0.0099	0.0068	0.0179	0.0189
0.06	0.0180	0.0055	0.0038	0.0053	0.0094	0.0077	0.0206	0.0116
0.07	0.0122	0.0069	0.0035	0.0054	0.0119	0.0084	0.0217	0.0079
0.08	0.0106	0.0078	0.0047	0.0083	0.0132	0.0093	0.0206	0.0098
0.09	0.0109	0.0089	0.0064	0.0089	0.0140	0.0093	0.0165	0.0104
0.10	0.0109	0.0089	0.0064	0.0089	0.0140	0.0093	0.0165	0.0104
σ_{t}	Runtime (s)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	280.3	131.2	75.7	49.0	39.4	26.9	23.3	22.9
0.03	279.6	132.2	75.5	49.0	39.5	28.1	23.6	23.1
0.04	275.5	129.9	77.4	50.2	33.2	23.4	23.8	22.7
0.05	280.9	129.4	76.4	49.5	32.8	21.5	25.0	21.8
0.06	191.3	127.7	76.5	49.3	33.4	26.9	23.6	22.9
0.07	282.4	127.8	77.2	46.1	38.6	24.0	23.9	23.6
0.08	283.6	127.4	70.8	50.2	38.3	27.2	23.4	23.9
0.09	278.8	127.5	76.7	49.4	38.9	27.8	24.1	23.5
0.10	278.1	127.6	76.6	49.1	37.8	26.9	24.5	24.1

Table 3 Results of GP registration in the Gazebo experiment

σ_t	Rotation error (rad)							
	$l = 0.3$ m	$l = 0.4$ m	$l = 0.5$ m	$l = 0.6$ m	$l = 0.7$ m	$l = 0.8$ m	$l = 0.9$ m	$l = 1.0$ m
0.02	0.0078	0.0065	0.0137	0.0110	0.0094	0.0099	0.0101	0.0079
0.03	0.0057	0.0083	0.0070	0.0052	0.0055	0.0078	0.0043	0.0047
0.04	0.0081	0.0067	0.0040	0.0025	0.0038	0.0032	0.0022	0.0025
0.05	0.0075	0.0035	0.0020	0.0015	0.0021	0.0019	0.0012	0.0014
0.06	0.0051	0.0021	0.0023	0.0014	0.0015	0.0021	0.0011	0.0014
0.07	0.0042	0.0017	0.0020	0.0013	0.0014	0.0022	0.0011	0.0015
0.08	0.0027	0.0015	0.0018	0.0014	0.0014	0.0022	0.0012	0.0014
0.09	0.0023	0.0014	0.0019	0.0014	0.0013	0.0022	0.0011	0.0015
0.10	0.0022	0.0014	0.0018	0.0013	0.0014	0.0022	0.0012	0.0016
σ_t	Translation error (m)							
	$l = 0.3$ m	$l = 0.4$ m	$l = 0.5$ m	$l = 0.6$ m	$l = 0.7$ m	$l = 0.8$ m	$l = 0.9$ m	$l = 1.0$ m
0.02	0.3593	0.3360	0.2374	0.2390	0.1854	0.2406	0.1845	0.1952
0.03	0.3584	0.2799	0.1365	0.1017	0.0940	0.1632	0.0663	0.1012
0.04	0.2983	0.1720	0.0749	0.0391	0.0637	0.0699	0.0278	0.0615
0.05	0.2511	0.0685	0.0417	0.0254	0.0321	0.0368	0.0153	0.0315
0.06	0.1816	0.0437	0.0277	0.0187	0.0244	0.0251	0.0124	0.0231
0.07	0.1559	0.0394	0.0245	0.0177	0.0233	0.0230	0.0106	0.0230
0.08	0.0986	0.0341	0.0233	0.0181	0.0247	0.0233	0.0131	0.0211
0.09	0.0768	0.0329	0.0219	0.0182	0.0245	0.0220	0.0129	0.0217
0.10	0.0726	0.0315	0.0233	0.0187	0.0255	0.0235	0.0145	0.0226
σ_t	Runtime (s)							
	$l = 0.3$ m	$l = 0.4$ m	$l = 0.5$ m	$l = 0.6$ m	$l = 0.7$ m	$l = 0.8$ m	$l = 0.9$ m	$l = 1.0$ m
0.02	2317.3	1030.8	459.5	271.9	187.0	128.9	98.5	77.7
0.03	2300.5	909.5	456.9	276.8	186.4	129.7	97.4	76.8
0.04	2068.0	905.0	455.2	278.4	186.7	131.3	99.2	76.8
0.05	2036.7	905.8	471.6	275.6	185.3	131.5	98.3	76.3
0.06	2045.0	887.1	476.4	276.1	186.1	130.0	98.7	78.1
0.07	2198.6	888.9	480.6	272.6	184.0	128.8	98.3	78.2
0.08	2137.1	895.1	483.6	272.8	184.3	133.6	98.4	77.3
0.09	2032.9	893.3	479.2	276.2	183.6	129.8	98.3	77.4
0.10	2058.0	901.1	481.9	276.1	185.0	129.0	98.5	77.7

discontinuities, the GP approach often produces obscure predictions at the boundary of the sub-regions, causing a reduction in accuracy.

2. As l increases, the computational time will dramatically decrease.

Theoretically, the computational time will decrease as l increases, as the larger sub-regions would contain more training points, causing the computational cost of calculating the inverse matrix in Eq. (2) to grow. However, with the utilization of the kd-tree-based acceleration technique, this cost is limited. In contrast, setting l to be small would cause the number of sub-regions to be enormous and would require an excessive number of for-loops in the procedure, causing a decrease in the computation speed.

3. The simpler and more structured the environment is, the better the performance of the method in terms of accuracy and efficiency.

A GP approach often needs more sub-regions to model an unstructured environment as compared to modeling a structured environment, as the point cloud measured from unstructured environments is more dispersed. Thus, the cost for modeling an unstructured environment is higher than that for modeling a structured environment. It is also harder for a GP to define a clear functional relationship for each sub-region in an unstructured environment. There are more incorrect predictive points generated when using GP to model an unstructured environment. Thus, the more unstructured the environment, the worse the transformation results.

The registration results for NDT and ICP from

Table 4 Results of GP registration in the Wood experiment

σ_t	Rotation error (rad)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	0.0217	0.0184	0.0210	0.0142	0.0179	0.0155	0.0243	0.0204
0.03	0.0108	0.0266	0.0289	0.0223	0.0256	0.0236	0.0258	0.0249
0.04	0.0188	0.0246	0.0249	0.0239	0.0261	0.0240	0.0266	0.0239
0.05	0.0210	0.0252	0.0255	0.0241	0.0263	0.0254	0.0260	0.0249
0.06	0.0218	0.0257	0.0259	0.0247	0.0263	0.0256	0.0261	0.0254
0.07	0.0250	0.0260	0.0261	0.0249	0.0264	0.0258	0.0263	0.0256
0.08	0.0233	0.0260	0.0262	0.0252	0.0265	0.0258	0.0266	0.0257
0.09	0.0246	0.0260	0.0262	0.0252	0.0266	0.0260	0.0266	0.0258
0.10	0.0242	0.0261	0.0261	0.0255	0.0267	0.0261	0.0268	0.0257
σ_t	Translation error (m)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	0.2393	0.0882	0.0731	0.0640	0.0600	0.1199	0.0623	0.0506
0.03	0.1365	0.0640	0.0653	0.0499	0.0690	0.0615	0.0806	0.0248
0.04	0.0628	0.0616	0.0629	0.0647	0.0790	0.0596	0.0765	0.0705
0.05	0.0444	0.0616	0.0698	0.0667	0.0796	0.0692	0.0741	0.0779
0.06	0.0445	0.0666	0.0739	0.0723	0.0785	0.0725	0.0753	0.0811
0.07	0.0444	0.0689	0.0773	0.0741	0.0774	0.0760	0.0785	0.0848
0.08	0.0495	0.0691	0.0767	0.0767	0.0788	0.0773	0.0804	0.0859
0.09	0.0601	0.0689	0.0778	0.0770	0.0802	0.0787	0.0817	0.0860
0.10	0.0557	0.0714	0.0795	0.0797	0.0815	0.0830	0.0833	0.0862
σ_t	Runtime (s)							
	$l=0.3$ m	$l=0.4$ m	$l=0.5$ m	$l=0.6$ m	$l=0.7$ m	$l=0.8$ m	$l=0.9$ m	$l=1.0$ m
0.02	4719.5	2053.0	1071.9	640.0	425.2	299.7	214.0	164.3
0.03	4805.1	2059.8	1073.7	641.6	427.8	303.7	214.4	164.1
0.04	4875.8	2062.6	1076.5	644.1	431.8	305.9	214.5	163.3
0.05	4795.8	2068.2	1081.0	644.0	428.7	303.5	214.5	164.5
0.06	4775.6	2067.3	1081.7	649.7	429.0	303.7	214.0	164.6
0.07	4776.1	2076.6	1085.2	648.8	428.7	302.3	214.4	164.5
0.08	4776.0	2080.3	1084.6	648.7	434.5	300.7	214.5	164.9
0.09	4789.8	2079.3	1089.0	647.5	432.5	303.6	213.3	165.0
0.10	4989.4	2083.5	1088.5	653.6	429.1	303.2	215.5	165.1

the three experiments are listed in Tables 5 (Stairs), 6 (Gazebo), and 7 (Wood). NDT was implemented with a different parameter l_n, ranging from 0.1 to 2.0 m. As NDT can easily converge to the wrong local minima, there are many results with extremely low accuracy. Following the same concept in Magnusson et al. (2007), we name these cases as failed cases and the implementations with acceptable results as good cases. All of the good cases are marked with boldface in these tables. As compared to these two methods, the advantage of our method with regard to accuracy is evident. The minimum rotation errors of our method are 0.0005, 0.0011, and 0.0108 rad for the Stairs, Gazebo, and Wood experiments, respectively. The rotation errors of NDT and ICP in the Stairs experiment are 0.0008 and 0.0110 rad, respectively. For the other two experiments, the accuracy of these two methods is even worse. All of the rotation errors from the good cases of NDT and ICP in the Gazebo and Wood experiments are larger than 0.05 rad. The situation for translation errors is almost the same. The minimum translation errors of our method are 0.0027, 0.0106, and 0.0248 m for the Stairs, Gazebo, and Wood experiments, respectively. The minimum translation error of the NDT method is 0.0034 m for the Gazebo experiment (l_n=0.4 m); however, except for that, the translation errors of NDT and ICP in all of the experiments are all larger than 0.02 m.

As for efficiency, our method has evident superiority as compared to NDT. The runtimes of our method reach approximately 23, 77, and 165 s

Table 5 Results of NDT and ICP in the Stairs experiment

Method	l_n (m)	Rotation error (rad)	Translation error (m)	Runtime (s)
NDT	0.1	0.0067	0.2692	289.8
	0.2	0.0356	0.2901	331.3
	0.3	0.0394	0.2823	272.6
	0.4	0.0120	0.2563	361.9
	0.5	0.0355	0.2666	457.3
	0.6	0.0330	0.3140	199.6
	0.7	0.0209	0.2269	197.4
	0.8	0.0438	0.2763	806.6
	0.9	0.0471	0.2907	293.9
	1.0	0.0037	0.1803	339.9
	1.1	0.0087	0.1253	154.6
	1.2	**0.0105**	**0.0560**	308.9
	1.3	0.0159	0.2165	435.4
	1.4	0.0159	0.4171	450.4
	1.5	**0.0008**	**0.0673**	549.0
	1.6	**0.0114**	**0.0266**	394.5
	1.7	**0.0032**	**0.0338**	230.0
	1.8	0.0541	0.2864	323.3
	1.9	0.0360	0.4218	245.8
	2.0	0.0488	0.2526	885.3
ICP	–	0.0110	0.0979	22.1

Good cases of NDT are in boldface

Table 6 Results of NDT and ICP in the Gazebo experiment

Method	l_n (m)	Rotation error (rad)	Translation error (m)	Runtime (s)
NDT	0.1	0.0572	0.2271	280.2
	0.2	0.0995	0.2798	286.2
	0.3	0.0088	0.3354	376.0
	0.4	**0.0674**	**0.0034**	322.7
	0.5	**0.0651**	**0.0945**	522.2
	0.6	0.0529	0.4074	492.2
	0.7	**0.0686**	**0.0534**	477.4
	0.8	0.0708	0.1152	276.7
	0.9	**0.0687**	**0.0771**	247.4
	1.0	**0.0682**	**0.0652**	395.6
	1.1	0.0686	0.1158	312.4
	1.2	**0.0655**	**0.0520**	547.4
	1.3	0.0660	0.4606	374.7
	1.4	**0.0666**	**0.0335**	310.6
	1.5	0.0688	0.1471	243.9
	1.6	0.0681	0.1095	154.4
	1.7	**0.0670**	**0.0569**	133.0
	1.8	**0.0650**	**0.0253**	103.9
	1.9	0.0702	0.2295	175.8
	2.0	**0.0660**	**0.0906**	131.1
ICP	–	0.0565	0.0566	13.3

Good cases of NDT are in boldface

for the Stairs, Gazebo, and Wood experiments, respectively. For NDT, these values are 154.6, 103.9, and 111.8 s. As a type of scan-to-scan registration method, the ICP method has the fastest computation speed. Nevertheless, the runtime of our method is almost equal to that of ICP in the Stairs experiment. This indicates that our method is competitive with ICP with regard to efficiency in certain structured environments.

Table 7 Results of NDT and ICP in the Wood experiment

Method	l_n (m)	Rotation error (rad)	Translation error (m)	Runtime (s)
NDT	0.1	0.1320	0.4903	296.0
	0.2	0.1386	0.4738	309.7
	0.3	0.1332	0.3713	451.7
	0.4	0.0759	0.3763	341.2
	0.5	0.1255	0.3850	361.4
	0.6	0.1195	0.3808	275.1
	0.7	0.1193	0.3857	364.1
	0.8	0.1170	0.3642	373.2
	0.9	0.0907	0.3385	141.4
	1.0	0.1040	0.2751	478.8
	1.1	0.1046	0.3228	182.3
	1.2	0.1000	0.3209	277.5
	1.3	0.0975	0.4096	253.6
	1.4	0.1044	0.1853	111.8
	1.5	0.1001	0.3134	146.1
	1.6	0.1025	0.2311	285.4
	1.7	0.0866	0.5666	207.8
	1.8	0.0955	0.3365	125.6
	1.9	**0.1049**	**0.1565**	226.6
	2.0	0.0984	0.2048	394.7
ICP	–	0.0984	0.1641	23.1

Good cases of NDT are in boldface

4.4 Discussion

As the goal of this study is to develop a scan-to-map point set registration method for mobile IUSs, NDT is our main comparison method. When applied to mobile IUSs, the robustness of the proposed method is an important performance index. As indicated from the results of NDT in Tables 5–7, the NDT method is sensitive to the selected parameters. There are only 4, 10, and 1 good cases out of 20 in the Stairs, Gazebo, and Wood experiments, respectively. The selection of parameters is rigorous when NDT is being applied. Meanwhile, there is a wide range of parameter choices for our method. As shown in Fig. 9, as long as parameters σ_t and l are not set too small or large, our registration can always result in a stable estimation.

As mentioned in Section 4.3, our method works well in a structured environment, but it is not very good for an unstructured environment. In that

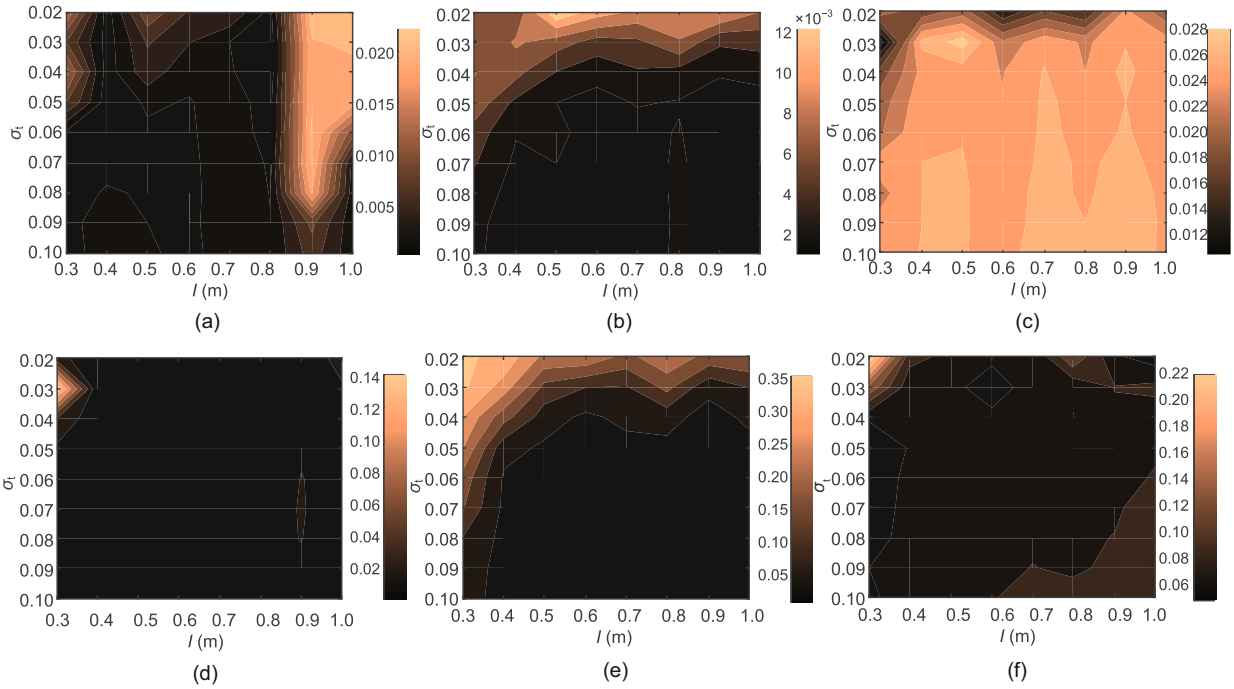

Fig. 9 Contour plots of the rotation errors in experiments Stairs (a), Gazebo (b), and Wood (c) and the translation errors in experiments Stairs (d), Gazebo (e), and Wood (f)

regard, NDT and ICP also fail to perform well in unstructured environments. It is difficult to find a specific value of l to make the proposed method work well in most scenarios. In practice, we choose l by rule of thumb. We admit that this is a major limitation of our method. When our method is applied to unstructured environments, we aim at the next best criterion, i.e., robustness. As discussed in the previous paragraph, our method is not sensitive to l, as shown in Fig. 9.

The results from the NDT approach reported in Magnusson et al. (2007) appear to be better than those from the ICP approach, which is different from the situation in this study. This is because in their study, the initial transformation was set as an adjustable parameter. Good results can be obtained with a small initial error. However, when a registration method is applied to mobile IUSs, the initial transformation is provided by initial sensors or wheel odometry and is not supposed to be adjustable. Accordingly, we did not set the initial transformation as an adjustable parameter for our research purposes.

The NDT approach can be accelerated by a certain type of down-sampling process, which is not used in this study. This may raise questions regarding whether this is a fair comparison, as the kd-tree acceleration used for GP map construction might be considered as a type of down-sampling process. Our answer is that kd-tree acceleration is fundamentally different from ordinary down-sampling processes. Using the nearest points of the test locations for calculation of the GP model barely affects the predictions at the test locations. However, the influence of the down-sampling process on the NDT approach with regard to accuracy can be relatively large. We prefer to considering our kd-tree acceleration technique as a search method, rather than a down-sampling method.

In the Gazebo and Wood experiments, the number of iterations for GP registration in all implementations reaches the maximum. A termination threshold of ΔT=0.001 seems to be too low for GP registration when it is applied to a more unstructured environment. We still set it at this value, so as to compare the performance of our method with a uniform standard.

5 Conclusions and future work

In this paper, we have introduced a novel 3D point set registration method based on a new type of map representation, namely, the GP map. Similar

to the point cloud map, the GP map (which is point-cloud-like) is compact and dense, and entails a lower memory cost for storage. Based on this new map representation, the registration problem has been handled analytically using an SVD method. The proposed method has been tested on three challenging real-world datasets, along with the ICP and NDT methods as benchmark algorithms. The experimental results showed that our method exhibits outstanding performance as compared to the other two widely used point-set-registration methods. We will extend this scan-to-map registration method to a complete SLAM framework in our future work.

Contributors

Bo LI and Yu ZHANG designed the research. Bo LI, Yu ZHANG, and Wen-jie ZHAO processed the data. Bo LI designed the computer programs and drafted the manuscript. Yu ZHANG, Wen-jie ZHAO, and Ping LI helped organize the manuscript. Bo LI, Yu ZHANG, and Wen-jie ZHAO revised and finalized the paper.

Compliance with ethics guidelines

Bo LI, Yu ZHANG, Wen-jie ZHAO, and Ping LI declare that they have no conflict of interest.

References

Besl PJ, McKay ND, 1992. A method for registration of 3-D shapes. *IEEE Trans Patt Anal Mach Intell*, 14(2):239-256. https://doi.org/10.1109/34.121791

Cadena C, Carlone L, Carrillo H, et al., 2016. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. *IEEE Trans Robot*, 32(6):1309-1332. https://doi.org/10.1109/TRO.2016.2624754

Chen Y, Medioni G, 1991. Object modeling by registration of multiple range images. Proc IEEE Int Conf on Robotics and Automation, p.2724-2729. https://doi.org/10.1109/ROBOT.1991.132043

Doherty K, Wang JK, Englot B, 2017. Bayesian generalized kernel inference for occupancy map prediction. Proc IEEE Int Conf on Robotics and Automation, p.3118-3124. https://doi.org/10.1109/ICRA.2017.7989356

Grisetti G, Kümmerle R, Stachniss C, et al., 2010. A tutorial on graph-based SLAM. *IEEE Intell Transp Syst Mag*, 2(4):31-43. https://doi.org/10.1109/MITS.2010.939925

Guizilini V, Ramos F, 2019. Variational Hilbert regression for terrain modeling and trajectory optimization. *Int J Robot Res*, 38(12-13):1375-1387. https://doi.org/10.1177/0278364919844586

Hess W, Kohler D, Rapp H, et al., 2016. Real-time loop closure in 2D LIDAR SLAM. Proc IEEE Int Conf on Robotics and Automation, p.1271-1278. https://doi.org/10.1109/ICRA.2016.7487258

Hornung A, Wurm KM, Bennewitz M, et al., 2013. OctoMap: an efficient probabilistic 3D mapping framework based on octrees. *Auton Robot*, 34(3):189-206. https://doi.org/10.1007/s10514-012-9321-0

Kim S, Kim J, 2013. Continuous occupancy maps using overlapping local Gaussian processes. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.4709-4714. https://doi.org/10.1109/IROS.2013.6697034

Li B, Wang YQ, Zhang Y, et al., 2020. GP-SLAM: laser-based SLAM approach based on regionalized Gaussian process map reconstruction. *Auton Robot*, in press. https://doi.org/10.1007/s10514-020-09906-z

Magnusson M, Lilienthal A, Duckett T, 2007. Scan registration for autonomous mining vehicles using 3D-NDT. *J Field Robot*, 24(10):803-827. https://doi.org/10.1002/rob.20204

O'Callaghan ST, Ramos FT, 2012. Gaussian process occupancy maps. *Int J Robot Res*, 31(1):42-62. https://doi.org/10.1177/0278364911421039

Plagemann C, Kersting K, Burgard W, 2008. Nonstationary Gaussian process regression using point estimates of local smoothness. Proc European Conf on Machine Learning and Knowledge Discovery in Databases, p.204-219. https://doi.org/10.1007/978-3-540-87481-2-14

Pomerleau F, Liu M, Colas F, et al., 2012. Challenging data sets for point cloud registration algorithms. *Int J Robot Res*, 31(14):1705-1711. https://doi.org/10.1177/0278364912458814

Rasmussen CE, Williams CKI, 2006. Gaussian Processes for Machine Learning. The MIT Press, Cambridge, USA.

Saarinen JP, Andreasson H, Stoyanov T, et al., 2013. 3D normal distributions transform occupancy maps: an efficient representation for mapping in dynamic environments. *Int J Robot Res*, 32(14):1627-1644. https://doi.org/10.1177/0278364913499415

Salvi J, Matabosch C, Fofi D, et al., 2007. A review of recent range image registration methods with accuracy evaluation. *Image Vis Comput*, 25(5):578-596. https://doi.org/10.1016/j.imavis.2006.05.012

Shen YR, Ng AY, Seeger M, 2006. Fast Gaussian process regression using kd-trees. Proc Annual Conf on Neural Information Processing Systems, p.1225-1232.

Smith M, Posner I, Newman P, 2010. Efficient non-parametric surface representations using active sampling for push broom laser data. Proc Robotics: Science and Systems. https://doi.org/10.15607/RSS.2010.VI.027

Stoyanov T, Magnusson M, Andreasson H, et al., 2012. Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *Int J Robot Res*, 31(12):1377-1393. https://doi.org/10.1177/0278364912460895

Thrun S, Burgard W, Fox D, 2005. Probabilistic Robotics. The MIT Press, Cambridge, USA.

Vasudevan S, Ramos F, Nettleton E, et al., 2009. Gaussian process modeling of large scale terrain. Proc IEEE Int Conf on Robotics and Automation, p.1047-1053. https://doi.org/10.1109/ROBOT.2009.5152677