IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

ID(O): Mapping Data Quantization for Bathymetric Collaborative SLAM

Qianyi Zhang, *Graduate Student Member, IEEE*, and Jinwhan Kim, *Member, IEEE*

***Abstract*—Underwater acoustic communication, characterized by limited bandwidth, high latency, and low reliability, poses significant challenges for data exchange in bathymetric collaborative simultaneous localization and mapping (CSLAM). In this article, we introduce a novel vector quantization (VQ) method called ID(O) for mapping data compression in bathymetric CSLAM. ID(O) encodes the map into an index map ($\mathbb{I}$), a central depth map ($\mathbb{D}$), and an orientation map ($\mathbb{O}$). To accommodate strict communication constraints, orientations can be partially or fully excluded from transmission, and we propose a method to estimate these orientations during map restoration. Moreover, we integrate ID(O) within a feature-based bathymetric CSLAM framework named TTT CSLAM. Extensive experiments on two large-scale sea trial datasets demonstrate that ID(O) achieves about 40% higher restoration accuracy than the baseline method using principal component analysis. TTT CSLAM with ID(O) can match that with lossless compression regarding mapping accuracy and efficiency, and it is robust against 40% packet loss and large dead reckoning drift errors across diverse environments. To the best of the authors' knowledge, ID(O) is the first VQ method for bathymetric data compression, and TTT CSLAM with ID(O) is the first bathymetric CSLAM tested within an underwater communication network employed by acoustic modems.**

***Index Terms*—Autonomous vehicle navigation, bathymetric navigation, marine robotics, multirobot systems, simultaneous localization and mapping (SLAM).**

I. Introduction

AUTONOMOUS underwater vehicles (AUVs) equipped with multibeam echosounders (MBES) have been widely employed to collect high-resolution seabed maps [1], [2]. However, dead reckoning (DR) drift errors, limited vehicle endurance, and restricted sensor measurement range make large-scale mapping challenging by a single AUV [3], [4]. Alternatively, bathymetric collaborative simultaneous localization and mapping (CSLAM) can be implemented by multiple vehicles to map an unknown region while simultaneously estimating their poses within the map [5].

Received 5 August 2025; accepted 4 November 2025. Date of publication 9 December 2025; date of current version 8 January 2026. This article was recommended for publication by Associate Editor K. Khosoussi and Editor J. Civera upon evaluation of the reviewers' comments. *(Corresponding author: Jinwhan Kim.)*

Qianyi Zhang is with the Robotics Program, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea (e-mail: qianyizhang@kaist.ac.kr).

Jinwhan Kim is with the Robotics Program, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea, and also with the Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea (e-mail: jinwhan@kaist.ac.kr).

Digital Object Identifier 10.1109/TRO.2025.3641755

Satisfying communication constraints is critical when designing CSLAM [6]. Despite its remarkable success on ground and aerial vehicles (e.g., [7], [8], [9], and [10]), conducting CSLAM using AUVs remains challenging due to the unique characteristics of underwater acoustic communication, including limited data rates (typically < 2000 bps), high latency (sound travels at about 1500 m/s), low reliability (20% to 60% packet loss), and reduced bandwidth (normally half-duplex systems) [6], [11]. Consequently, reducing the frequency of data exchange and data volume transmitted between vehicles to match the capability of acoustic communication becomes paramount for bathymetric CSLAM.

As a result, due to the high latency and packet loss caused by multiple data exchanges via acoustic channels during distributed loop closure detection (transmitting local and global descriptors) and distributed pose graph optimization (swapping graph updates), bathymetric CSLAM is commonly implemented in a centralized configuration rather than a distributed architecture (e.g., [5] and [12]). Moreover, unlike the typical visual-based or LiDAR-based methods, existing bathymetric CSLAM methods generally transmit mapping data rather than landmark-based graphs (e.g., [13]), global descriptors or features (e.g., [7] and [9]) among vehicles because the following holds.

1) Evaluating depth differences between submaps is crucial for loop closure detection and data association in bathymetric SLAM (e.g., [14] and [15]).
2) Transmitting global descriptors for loop closure detection followed by sending submaps for registration requires multiple acoustic communications, causing high latency particularly in long-range scenarios.
3) Although features and global descriptors have been proposed for bathymetric SLAM in [16] and [17], these dense representations require more bandwidth than directly sending compressed submaps.

Therefore, mapping data compression becomes a bottleneck in bathymetric CSLAM. However, the existing methods, such as image-based (e.g., portable network graphics (PNG) [18]), data-structure-based (e.g., quad-tree [19]), and regression-based (e.g., Gaussian process regression [12]) methods struggle to preserve map details at high compression ratios (CR). Moreover, existing bathymetric CSLAM heavily relies on brute-force matching for loop closure detection, significantly increasing the computational costs during large-scale missions.

Vector quantization (VQ) has been successfully applied to underwater image compression (e.g., [20] and [21]). It involves constructing a codebook, and each input vector can

be represented by its closest word in the book. Therefore, high-dimensional input data can be encoded into a finite set of scalar indexes of the words [22]. Theoretically, it can achieve lossless compression at high CR if the words in the codebook perfectly match the input vectors. However, its implementation to bathymetric data has yet to be realized due to the difficulty of constructing a codebook that can be generalized across diverse scenarios. Moreover, training the codebook using prior knowledge of the environment is also infeasible for CSLAM missions.

In previous work, a feature-based bathymetric SLAM framework named TTT SLAM was introduced in [16], where the acronym "TTT" represents its three main components: terrain gradient features, TEASER++ registration [23], and graduated nonconvexity truncated least squares (GNC-TLS) robust backend [24]. Following this, a generalized bag-of-words (BoW) called Shape BoW was proposed for realizing appearance-based loop closure detection in bathymetric SLAM [17]. We reveal that TTT SLAM with Shape BoW offers advantages as the backbone for bathymetric CSLAM because the following holds.

1) Its appearance-based loop closure detection relies solely on comparing map indexes, which is significantly faster than brute-force matching.
2) Its data association is based on TEASER++, a registration method without the need for an initial guess.
3) Shape BoW captures the common patterns of local seabed topography, making it an ideal foundation for constructing a generalized VQ codebook.
4) TTT SLAM has shown superior performance, particularly in robustness and efficiency, compared to the existing methods (see [16] and [17]).

In this study, we propose a mapping data quantization method capable of preserving map details at high CR and integrate it with a feature-based bathymetric CSLAM framework. The contributions of this work are summarized as follows.

1) As illustrated in Fig. 1, we introduce a novel VQ method called ID(O) for bathymetric data compression. The VQ codebook is constructed by extending Shape BoW through data augmentation. Using this codebook, a bathymetric map can be compressed into an index map ($\mathbb{I}$), a central depth map ($\mathbb{D}$), and an orientation map ($\mathbb{O}$).
2) To accommodate strict communication constraints, orientation information can be partially or fully excluded from transmission, which is the basis for the name "ID(O)"—indicating orientations are optional. Alternatively, we propose a method for estimating $\mathbb{O}$ during the restoration phase by minimizing the consistency errors of the restored map.
3) To demonstrate the applicability and performance of ID(O) in CSLAM, we integrate it within the multiagent extension of TTT SLAM (named TTT CSLAM). The Shape BoW used in TTT CSLAM is naturally utilized as the basis for VQ codebook construction in ID(O).
4) Extensive experiments were conducted using two large-scale sea trial datasets, including the comparison of ID(O) against six existing compression methods, evaluation of TTT CSLAM with different compression methods under varying packet loss and DR noise levels, and the test of TTT CSLAM with ID(O) in an underwater communication network called Unetstack [25] under an extremely low data rate (200 bps). To the best of our knowledge, this is the first time a bathymetric CSLAM was tested within a communication network employed by acoustic modems. The results demonstrated that ID(O) achieves 40% higher restoration accuracy compared to the best baseline method using principal component analysis (PCA). Moreover, TTT CSLAM with ID(O) can match that with lossless compression regarding mapping accuracy and efficiency, and it is robust to 40% packet loss and significant DR drift errors across diverse environments.

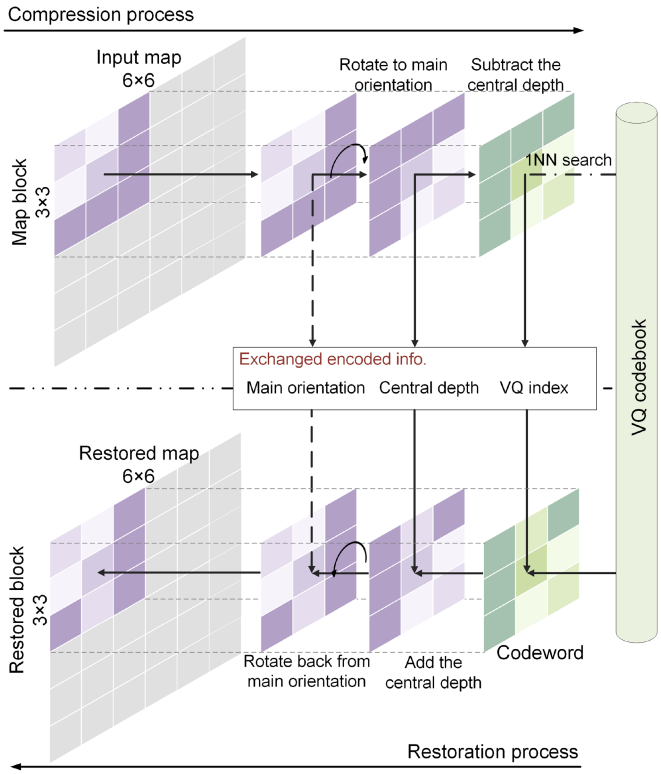

Fig. 1. Example of the compression and restoration of a map block using ID(O). The color variations in the block represent the depth differences. The block is encoded to an orientation (optional, marked by dashed lines), a depth, and an index. The VQ codebook is stored in all vehicles before the mission.

The rest of this article is organized as follows. Section II reviews related work. Section III details the generation of a VQ codebook. Sections IV and V introduce the compression and restoration processes of ID(O), respectively. Section VI explains the method for estimating the orientation information. Section VII presents TTT CSLAM framework. Section VIII demonstrates experimental results on two sea trial datasets. Finally, Section IX concludes this article.

II. Related Work

A. Bathymetric Data Compression

Bathymetric data compression is typically designed to reduce the data volume of large-scale, high-resolution bathymetric

maps to address storage constraints [26]. Due to the similarity between these 2.5D maps and images, image compression methods, such as PNG, joint photographic experts group (JPEG), and JPEG 2K, are widely implemented as their convenience and viability in geospatial data processing software [27]. However, these methods struggle to achieve high CR [28]. Therefore, specialized bathymetric data compression methods have been developed to minimize data size while maintaining the differences between the original and restored maps below predefined thresholds [28]. For instance, the methods using discrete cosine transform (DCT) and wavelet transform detailed in [28] and [29] adjust the number of DCT and wavelet coefficients in each map block to achieve desired restoration accuracy. A similar method proposed in [30] leverages PCA with a prestored eigensurface database, achieving superior performance compared to the methods based on DCT and wavelet transforms in the experiment.

There are also works focusing on minimizing the data volume transmitted between AUVs, which are more relevant to this research. Sparse Gaussian process (SGP) regression has shown promise for bathymetric map modeling (e.g., [31] and [32]), and its first application to mapping data compression in bathymetric CSLAM was introduced in [12], where a set of inducing points (a minimum of 120 points as suggested by the authors) and model parameters are transmitted between vehicles for map restoration. While this approach effectively preserves general map structures, it struggles to capture fine details when inducing points are limited, and training regression models during missions is computationally expensive [5]. Alternatively, the method described in [33] decomposes grid maps into blocks using a quad-tree structure, but this method sacrifices local details and introduces blocking artifacts under high CR, resulting in invalid terrain gradient features extracted along block boundaries.

Beyond the techniques discussed above, learning-based approaches demonstrate the advantages in image and 3D point cloud compression (see surveys in [34] and [35], and [36] and [37], respectively), and the extension of these methods to the underwater domain (primarily for underwater image compression) has emerged recently (e.g., wavelet transform with convolutional neural network [38], and autoencoder [39], [40]). However, network training relies on large datasets containing representative data distributions [41]. Due to the lack of suitable open bathymetric datasets for training, the potential generalization problem of the model to unknown environments, and the limited computational resources (such as high-performance graphics processing units) on AUVs, the successful deployment of these methods in bathymetric CSLAM remains an open problem.

B. CSLAM on Aerial and Ground Vehicles

Extensive CSLAM methods have been developed for aerial and ground vehicles (e.g., [42], [43], [44], and [45]). Centralized CSLAM relies on a central node to aggregate information from all vehicles and handle critical tasks, such as loop closure detection, data association, and graph optimization, while client vehicles maintain only essential modules, such as visual odometry. Examples of this configuration can be found in [46], [47], [48], [49], and [50].

In contrast, distributed CSLAM does not require a central node, thereby improving the scalability and robustness of the system [51]. These methods utilize distributed loop closure detection and distributed pose graph optimization. At the frontend, global descriptors (e.g., BoW indexes [52], NetVLAD descriptors [53], or scan context descriptors [54]) are exchanged between vehicles for place recognition [55]. Complementary detection methods considering communication constraints have been proposed and evaluated on visual datasets in [56]. At the backend, an early work employs Gaussian elimination to construct condensed graphs for transmission [57], with its advanced version in [13] introducing anti-factors to address the double counting problem effectively. Distributed Gauss–Seidel-based methods have also been implemented, such as in [7] and [58]. More recently, Nesterov's acceleration is introduced in [59] to boost the optimization process. Moreover, a Riemannian block coordinate descent solver with distributed global optimality verification proposed in [60] achieves a similar acceleration, and its robust extension is utilized as the backend of a distributed CSLAM framework named Kimera-Multi [10]. A comprehensive review of existing CSLAM methods can be found in [61].

C. Underwater Collaborative Navigation

In multi-AUV systems, ranging information between vehicles derived from acoustic communication can be utilized for cooperative localization (CL). These methods are commonly based on Bayesian approaches, such as Kalman filtering [62], [63], [64] or belief propagation [65]. The performance of CL can also be enhanced by integrating terrain-aided navigation techniques (e.g., [66] and [67]). However, the commonly used one-way travel time ranging method requires clock synchronization between AUVs for distance measurement, but asynchronization caused by clock drift is inevitable, particularly during long-term underwater navigation [68]. Alternatively, two-way travel time ranging can be adopted, but the high latency of acoustic communication degrades the measurement frequency, and the relative motion of the vehicles introduces additional distance errors [68], [69], [70].

While ranging information can be utilized, underwater CSLAM primarily relies on data collected by perception sensors for navigation in unknown regions. This remains an open topic due to stringent acoustic communication constraints.

Early studies explored the use of imaging information. For example, in [71], underwater images are exchanged between vehicles only when their Euclidean distance (calculated from the graph) falls below a threshold to reduce the data volume for transmission. However, this odometry-based loop closure detection is less effective under significant DR drift errors. This method is also extended to CSLAM with 3-D imaging sonars in [72], where a plane-based registration method is implemented for data association. Nonetheless, loop closures are manually specified in this research. McConnell et al. [73] transmited global descriptors for loop closure detection and employed a simple voxel-based downsampling method to compress sparse data from imaging sonars, with each vehicle maintaining its own SLAM solution. However, downsampling (e.g., map gridding)

is typically a preprocessing step for handling large-scale dense bathymetric data (see Section II-A), and additional processes are commonly required for further compression. CSLAM using side-scan sonars has also been investigated. For example, in [6], the intermediate poses between the contact points of the vehicles are marginalized, and graph sparsification is performed to create condensed landmark-based graphs for transmission.

Limited research has addressed CSLAM using MBES, and these methods employ centralized configurations with a focus on mapping data compression due to the strict acoustic communication constraints outlined in Section I. For example, in [12], maps are compressed using SGP regression, and loop closure detection with data association is conducted by template matching. However, this method is computationally intensive and limited to reducing horizontal position errors. To overcome this, template matching is replaced by a submap registration method based on genetic algorithm (GA) in [5], but brute-force matching is still required to match every submap within a pose uncertainty ellipse against the received submap using computationally expensive GA.

From the papers introduced above, existing bathymetric data compression methods struggle to preserve details at high CR, and bathymetric CSLAM heavily relies on brute-force matching at the frontend. In this work, we propose a novel mapping data quantization method capable of retaining map details at high CR and integrate it within a feature-based bathymetric CSLAM framework enabling effective loop closure detection and data association.

III. Generating a VQ Codebook

A. Overview of Related Terminology

Several concepts from previous works [16], [17] are utilized in this article, with brief definitions provided below.

1) *Terrain gradient features:* These handcrafted features are extracted from the locations with high terrain standard deviations (TSD) in a gridded bathymetric map using a sliding window [74]. The descriptors of the features are defined as the local maps surrounding the feature locations, and these descriptors are aligned to their main orientations for rotational invariance. Because of the direct utilization of map blocks, these descriptors contain both relative depth information (shapes) and absolute depth information (where the shapes are located).
2) *Main orientation:* It represents the dominant direction of local slopes within a descriptor (map), indicating its facing direction. Given a descriptor, the orientations of the grids within the descriptor are calculated by computing surface normals, and these orientations are grouped using a histogram with 36 bins, where each bin represents a 10° range of orientations. The main orientation of the descriptor is determined by fitting a quadratic curve to the highest bin and its adjacent bins.
3) *Shape BoW:* It is a generalized BoW containing a series of local terrain shapes for appearance-based loop closure detection in bathymetric SLAM. Shape BoW is constructed by clustering the shapes of terrain gradient feature descriptors, as these shapes follow common patterns due to the smoothing of ocean currents.

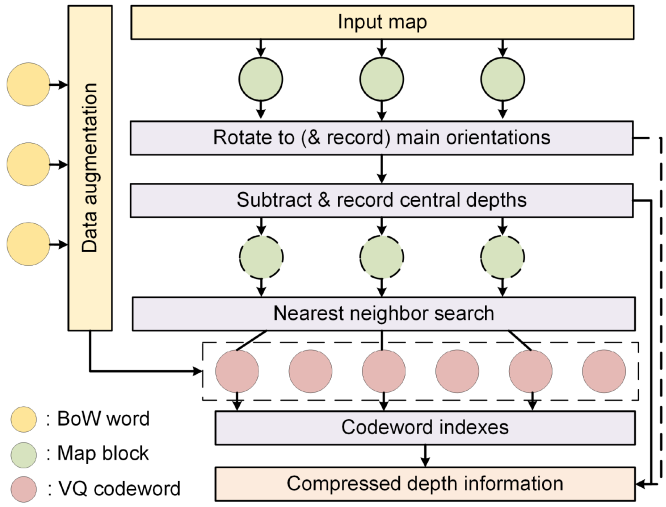

Fig. 2. Example of the depth encoding process of ID(O). The map blocks with dashed lines represent the blocks with only shape information. The VQ codebook is constructed by data augmentation before the mission. For an easy illustration, the size of Shape BoW and the number of map blocks are three.

B. Construction of a VQ Codebook

A VQ codebook is typically constructed by clustering the training data exhibiting the same patterns as input data. Shape BoW is an ideal basis for codebook construction, as it contains general shapes of the local seabed topography. However, since Shape BoW is trained from the features extracted from the regions with high TSD values, it may not generalize well to flat regions. Therefore, data augmentation is conducted to Shape BoW for codebook construction.

Given a Shape BoW (marked as $\mathbb{W}$) storing W terrain shapes (shape size: $J \times J$), a VQ codebook (denoted as $\mathbb{V}$) is defined as a collection of $\mathbb{W}$ with a finite set of distinct scaling factors

$$\mathbb{V} = \left\{ k_i \mathbb{W} \mid k_i \in \mathbb{R}^+, i = 1, 2, \ldots, \mathcal{I} \right\} \tag{1}$$

where k_i is the ith scaling factor, and $\mathcal{I}$ is the total number of these factors. By adjusting k_i, subdued ($0 < k_i < 1$) or steep ($k_i > 1$) shapes can be generated to enrich the codebook.

IV. Compression of Bathymetric Data

By utilizing the codebook, ID(O) encodes the dense depth information of a map into three low-dimensional representations: codebook indexes, central depths, and main orientations. Considering an $a \times b$ size input grid map $\mathbb{M}$ containing both horizontal and depth information, the former is encoded by storing the 2-D coordinates of the bottom-left grid $p(x_0, y_0)$ (float) along with the grid resolution r (int8).

An example of the depth encoding process is demonstrated in Fig. 2. The input map $\mathbb{M}$ is divided into N nonoverlapping

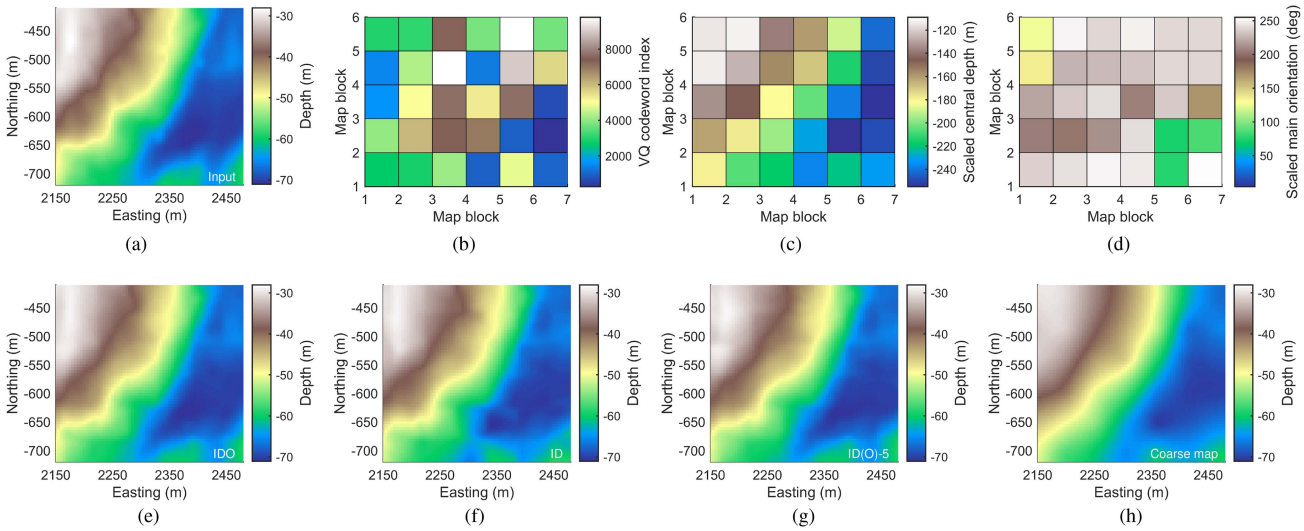

Fig. 3. Example of using ID(O) for bathymetric map compression. (a) Input map (after imputation). (b) Index map ($\mathbb{I}$). (c) Central depth map ($\mathbb{D}$, we reverse the values for a better illustration). (d) Main orientation map ($\mathbb{O}$). (e) Restored map using both $\mathbb{I}$, $\mathbb{D}$, and $\mathbb{O}$ (IDO). (f) Restored map using only $\mathbb{I}$ and $\mathbb{D}$ (ID). (g) Restored map using $\mathbb{I}$ and $\mathbb{D}$ with five essential main orientations [ID(O)-5]. (h) Coarse map constructed using only $\mathbb{D}$.

$J \times J$ size blocks

$$\mathbb{M} = \{M_i\}_{i=1}^{N} \tag{2}$$

where

$$N = \lfloor a/J \rfloor \times \lfloor b/J \rfloor \tag{3}$$

M_i is the ith map block, and $\lfloor \cdot \rfloor$ is the floor operation. The size of M_i matches that of the words in the codebook, which is also equivalent to the size of terrain gradient feature descriptors.

Subsequently, the main orientation of each block is calculated using an orientation histogram and recorded for transmission. Each block is then aligned to its main orientation accordingly

$$\tilde{M}_i = R(\theta_i^{\mathrm{m}}) M_i \tag{4}$$

where $\tilde{M}_i$ represents the orientation-aligned block, and $R(\theta_i^{\mathrm{m}})$ is the rotation matrix corresponding to the main orientation θ_i^{m}.

Following this, the absolute depth information is recorded and removed from each block by central depth subtraction

$$\hat{M}_i = \tilde{M}_i - m_i^{\mathrm{c}} \tag{5}$$

where $\hat{M}_i$ and $m_i^{\mathrm{c}} \in M_i$ are the depth-normalized block and the central depth value of M_i, respectively.

Finally, for each $\hat{M}_i$, the index of its nearest word (marked as I_i) in the codebook $\mathbb{V}$ is searched by minimizing

$$I_i = \arg\min_{V_{I_i} \in \mathbb{V}} \frac{1}{J^2} \sum_{k=1}^{J^2} \left(\hat{m}_i(k) - v_{I_i}(k)\right)^2 \tag{6}$$

where $V_{I_i} \in \mathbb{V}$, $\hat{m}_i(k) \in \hat{M}_i$, and $v_{I_i}(k) \in V_{I_i}$ represent the nearest neighbor of $\hat{M}_i$ in $\mathbb{V}$, the kth element of $\hat{M}_i$, and the kth element of V_{I_i}, respectively.

As a result, ID(O) encodes each $J \times J$ size map block M_i into only three scalars: a VQ index I_i, a center depth m_i^{c}, and a main orientation θ_i^{m}. In other words, the depth information of a map can be compressed to an index map $\mathbb{I}$, a central depth map $\mathbb{D}$, and an orientation map $\mathbb{O}$

$$\mathbb{I} = \{I_i\}_{i=1}^{N}, \quad \mathbb{D} = \{m_i^{\mathrm{c}}\}_{i=1}^{N}, \quad \mathbb{O} = \{\theta_i^{\mathrm{m}}\}_{i=1}^{N} \tag{7}$$

where all these maps have a size of $\lfloor a/J \rfloor \times \lfloor b/J \rfloor$. An example of these maps is demonstrated in Fig. 3(b)–(d).

In application, $\mathbb{I}$ is stored in unsigned int16 format (assuming the size of the VQ codebook fits within the format range), while $\mathbb{D}$ and $\mathbb{O}$ are scaled and stored in unsigned int8 format to further reduce the packet size

$$\mathbb{D} = \lfloor f_d \times \mathbb{D} \rceil, \quad \mathbb{O} = \lfloor f_o \times \mathbb{O} \rceil \tag{8}$$

where $\lfloor \cdot \rceil$ represents the rounding operation, $f_d = 255/d_{\min}$ and $f_o = 255/\theta_{\max}$ are scaling factors, $d_{\min}$ is the minimum depth in $\mathbb{D}$, and $\theta_{\max}$ is the maximum angle in $\mathbb{O}$. Therefore, the acoustic packets generated by ID(O) consist of $\mathbb{I}$ (index map), $\mathbb{D}$ (central depth map), $\mathbb{O}$ (main orientation map), f_d (depth scaling factor), f_o (orientation scaling factor), $p(x_0, y_0)$ (bottom-left grid position), and r (grid resolution).

V. Restoration of Bathymetric Data

The restoration process of ID(O) involves recovering the horizontal and depth information of the map. The coordinates of a grid position $p(x_i, y_j)$ are calculated as

$$p(x_i, y_j) = (x_0 + i \cdot r, y_0 + j \cdot r) \tag{9}$$

where $i \in [0, \lfloor a/J \rfloor J)$ and $j \in [0, \lfloor b/J \rfloor J)$ are the indices of the grid along the x- and y- directions, respectively.

As demonstrated in Fig. 4, the restoration of depth information follows the inverse operations of the compression. Considering a VQ index I_i of a block, its codeword V_{I_i} is used as the shape of the restored block. Subsequently, the absolute depth information is added back

$$\breve{M}_i = V_{I_i} + m_i^{\mathrm{c}}/f_d \tag{10}$$

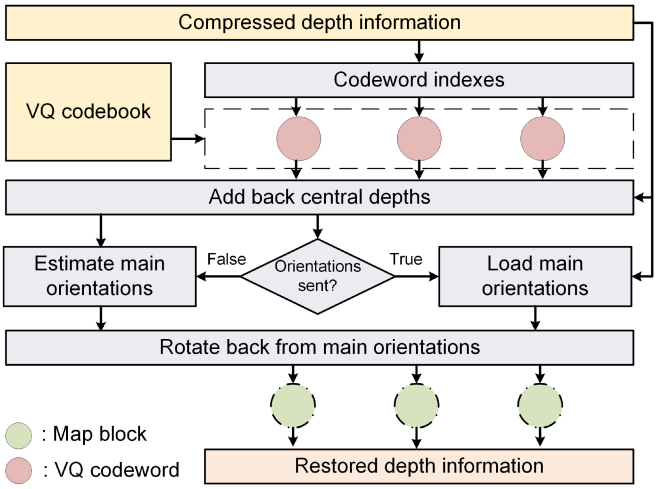

Fig. 4. Example of the depth restoration process of ID(O). The map blocks with dashed lines represent the blocks restored using ID(O).

where $\check{M}_i$ is the block i with only absolute depth information restored. Following this, $\check{M}_i$ is rotated back to its original orientation using θ_i^{m}

$$M_i' = R^T(\theta_i^{\mathrm{m}}/f_o)\check{M}_i \tag{11}$$

where M_i' is the restored map block i from the packet.

Due to the rotation of the codewords, there are missing values within the restored map $\mathbb{M}' = \{M_i'\}_{i=1}^N$. Therefore, an imputation method utilizing penalized least square regression based on 3-D discrete cosine transform (DCT-PLS) [75] is applied to fill the missing values in $\mathbb{M}'$. Moreover, Gaussian filtering is used to smooth the map and reduce the inconsistency at the boundaries of adjacent map blocks. As shown in Fig. 3(e), the map details can be effectively preserved by using ID(O).

VI. Estimation of Main Orientations

As explained in Section V, $\mathbb{I}$, $\mathbb{D}$, and $\mathbb{O}$ are required for restoring maps. However, we reveal there is redundancy between $\mathbb{O}$ and the other two maps. As a result, $\mathbb{O}$ can be excluded from transmission to further reduce the packet size, and we propose a method to estimate $\mathbb{O}$ instead. As illustrated in Algorithm 1, this is achieved by minimizing a cost function consisting of the mean squared error (MSE) between the boundaries of adjacent blocks (marked as E^{b}) and that between the restored map and a coarse map constructed by using $\mathbb{D}$ (marked as E^{r}).

A. Calculating Consistency Errors Between Adjacent Blocks

After applying (10) to each block, padding is conducted on each $\check{M}_i$ to expand its size by adding a single row (column) on both sides, with the values of the padded grids calculated using DCT-PLS imputation. As demonstrated in Fig. 5, padding creates two-row (column) overlapping regions between adjacent blocks.

Given a padded block $\check{M}_i$ rotated by an estimated angle θ_i^{e}

$$M_i' = R^T(\theta_i^{\mathrm{e}})\check{M}_i \tag{12}$$

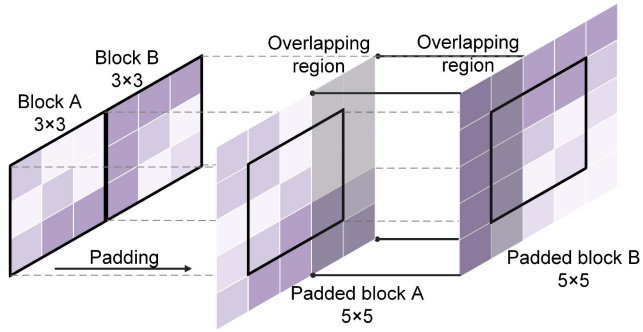

Fig. 5. Padding is conducted to the blocks to create an overlapping region (marked in gray). We compute the MSE between the blocks within this region to evaluate their consistency.

Algorithm 1: Estimation of Main Orientations.

Input: Index map $\mathbb{I}$, depth map $\mathbb{D}$ with scaling factor f_d, VQ codebook $\mathbb{V}$, empty codeword V^{NaN};
Output: Estimated main orientations $\mathbb{O}^*$;

Initialization: $E^{\mathrm{b}} = 0$, $E^{\mathrm{r}} = 0$, $\mathbb{M}^{\mathrm{r}} = \{M_i^{\mathrm{r}}\}_{i=1}^N = \emptyset$
for $i = 1$ to N **do**
  $\check{M}_i = V_{I_i} + m_i^{\mathrm{c}}/f_d$
  $\check{M}_i = \mathrm{padding}(\check{M}_i)$
  $M_i' = R^T(\theta_i^{\mathrm{e}})\check{M}_i$
  $M_i^{\mathrm{r}} = V^{\mathrm{NaN}} + m_i^{\mathrm{c}}/f_d$
end for
$\mathbb{M}^{\mathrm{r}} = \mathrm{imputation}(\mathbb{M}^{\mathrm{r}})$
for $i = 1$ to N **do**
  for $j = \mathrm{left}(i)$ and $\mathrm{top}(i)$ **do**
    $e_{i,j}^{\mathrm{b}} = \frac{1}{|S|}\sum_{k\in S}(m_i'(k) - m_j'(k))^2$
  end for
  $E^{\mathrm{b}} \mathrel{+}= \delta_i^{\mathrm{left}} \cdot e_{i,\mathrm{left}(i)}^{\mathrm{b}} + \delta_i^{\mathrm{top}} \cdot e_{i,\mathrm{top}(i)}^{\mathrm{b}}$
  $E^{\mathrm{r}} \mathrel{+}= \frac{1}{J^2}\sum_{k=1}^{J^2}(m_i^{\mathrm{r}}(k) - m_i'(k))^2$
end for
$\mathbb{O}^* = \arg\min(E^{\mathrm{b}} + E^{\mathrm{r}})$
Return $\mathbb{O}^*$

the MSE between the boundary of this block and that of the adjacent block (M_j') is calculated

$$e_{i,j}^{\mathrm{b}} = \frac{1}{|S|}\sum_{k\in S}\left(m_i'(k) - m_j'(k)\right)^2 \tag{13}$$

where S and $|S|$ represent the set of overlapping grids between the blocks and their total number respectively, and $m_i'(k)$ represents the grid depth at position k in M_i'. For each block, only the errors with its left and top adjacent blocks are calculated to avoid redundant computation (The error is set to 0 if no adjacent block exists in that direction). Therefore, E^{b} can be expressed as

$$E^{\mathrm{b}} = \sum_{i=1}^{N}\left(\delta_i^{\mathrm{left}} \cdot e_{i,\mathrm{left}(i)}^{\mathrm{b}} + \delta_i^{\mathrm{top}} \cdot e_{i,\mathrm{top}(i)}^{\mathrm{b}}\right) \tag{14}$$

where δ_i^{dir} is an indicator function that equals 1 if block i has an adjacent block in the corresponding direction, or 0 otherwise,

and dir(i) represents the index of the adjacent block in that direction.

B. Calculating Consistency Errors Between Restored and Coarse Map Blocks

To simplify the illustration, we define an empty codeword V^{NaN}, where all values are empty (NaN) except the central value is zero. We replace the VQ codeword V_{I_i} in (10) by V^{NaN}

$$M_i^{\text{r}} = V^{\text{NaN}} + m_i^{\text{c}}/f_d \tag{15}$$

where M_i^{r} is a map block containing only central depth information. The set $\mathbb{M}^{\text{r}} = \{M_i^{\text{r}}\}_{i=1}^{N}$ forms a map with the same resolution and size as the restored map but retains only the depths at the block center positions. Subsequently, we conduct DCT-PLS imputation to $\mathbb{M}^{\text{r}}$ to fill the empty values. As shown in Fig. 3(h), this map is constructed solely using $\mathbb{D}$, but it contains the coarse details of the map that can guide the optimization. The inconsistency between $\mathbb{M}^{\text{r}}$ and the restored map is calculated as

$$E^{\text{r}} = \frac{1}{J^2}\sum_{i=1}^{N}\sum_{k=1}^{J^2}\left(m_i^{\text{r}}(k) - m_i'(k)\right)^2 \tag{16}$$

where $m_i^{\text{r}}(k)$ and $m_i'(k)$ are the depth of the grid at position k in M_i^{r} and M_i', respectively (excluding the padding regions).

Finally, the optimization problem can be expressed as

$$\begin{aligned}\mathbb{O}^* &= \arg\min_{\mathbb{O}} \left(E^{\text{b}} + E^{\text{r}}\right) \\ &= \arg\min_{\mathbb{O}} \sum_{i=1}^{N}\Bigg[\left(\delta_i^{\text{left}}\cdot e_{i,\text{left}(i)}^{\text{b}} + \delta_i^{\text{top}}\cdot e_{i,\text{top}(i)}^{\text{b}}\right) \\ &\quad + \frac{1}{J^2}\sum_{k=1}^{J^2}\left(m_i^{\text{r}}(k) - m_i'(k)\right)^2\Bigg].\end{aligned} \tag{17}$$

This is a separable problem and should be solved within limited function evaluations for online implementation. We conducted the test to find the best optimizer for this problem, including the 25 methods listed in [76] and the methods in [77]. Ultimately, we selected the naive multiscale search optimization (NMSO) method designed to solve the problems within a limited budget [78]. NMSO explores the search space by hierarchically dividing it into multiple scales to find the optimal solution. It also provides consistency and a finite-time convergence rate. Details of NMSO can be found in [78].

After having $\mathbb{O}^*$, the procedure introduced in Section V is conducted to restore bathymetric maps. By excluding the transmission of orientations, ID(O) effectively reduces data size at the cost of increased computational time during restoration. Nevertheless, it is a reasonable tradeoff since acoustic communication resources are significantly more expensive than computational resources.

C. Selecting Essential Main Orientations

While most orientations can be accurately estimated, NMSO occasionally fails to determine the main orientations of some blocks due to the limited function evaluations on this high-dimensional problem. An example is shown in Fig. 3(f), where a map block in the bottom-right section is incorrectly aligned to another orientation, causing a noticeable inconsistency with the ground truth. To address this issue, we can transmit the orientations of such blocks (referred to as "essential main orientations") to enhance restoration accuracy and reduce the dimensionality of the problem (17), while the remaining orientations are estimated accordingly.

Algorithm 2: Selecting Essential Main Orientations.

Input: Input map $\mathbb{M}$, index map $\mathbb{I}$, depth map $\mathbb{D}$, orientation map $\mathbb{O}$, VQ codebook $\mathbb{V}$, the number of essential main orientations to select N_{ess};
Output: Selected orientations $\mathbb{O}_{\text{ess}}$;

Initialization: List of block errors $\mathbb{E} = \emptyset$
$\mathbb{O}^* = \text{orientationEstimation}(\mathbb{I}, \mathbb{D}, \mathbb{V})$
$\mathbb{M}' = \text{mapRestoration}(\mathbb{I}, \mathbb{D}, \mathbb{O}^*, \mathbb{V})$
for $i = 1$ to N **do**
  $E_i^{\text{block}} = \frac{1}{J^2}\sum_{k=1}^{J^2}(m_i(k) - m_i'(k))^2$
  $\mathbb{E} \leftarrow E_i^{\text{block}}$
end for
% Identify block indices with top N_{ess} highest errors
$I_{\text{top}} = \text{argsort}(\mathbb{E}, \text{descending})[1 : N_{\text{ess}}]$
% Extract orientations corresponding to I_{top}
$\mathbb{O}_{\text{ess}} = \{\theta_i^{\text{m}} \mid i \in I_{\text{top}}, \theta_i^{\text{m}} \in \mathbb{O}\}$
Return $\mathbb{O}_{\text{ess}}$

As outlined in Algorithm 2, the selection is achieved by evaluating the inconsistency between the input map blocks and the restored map blocks. During the compression phase, we additionally conduct the map restoration using only $\mathbb{I}$ and $\mathbb{D}$. The MSE between each block of the restored map (M_i') and the corresponding block of the input map (M_i) is calculated

$$E_i^{\text{block}} = \frac{1}{J^2}\sum_{k=1}^{J^2}\left(m_i(k) - m_i'(k)\right)^2 \tag{18}$$

where E_i^{block} is the MSE between the blocks. We select the orientations of the blocks with the top N_{ess} highest MSE as the essential main orientations (marked as $\mathbb{O}_{\text{ess}}$), and these orientations are scaled and stored for transmission. As shown in Fig. 3(g), preserving only the top five essential main orientations results in improved restoration accuracy.

VII. TTT CSLAM Framework

As shown in Fig. 6, TTT CSLAM preserves key components of TTT SLAM, with standard modifications for multiagent settings. It is in a centralized configuration, but each vehicle can also maintain a SLAM solution (e.g., [79] and [80]) if a suitable communication protocol is employed (e.g., [81]). Similar to most underwater SLAM and CSLAM methods (e.g., [2], [5], and [6]), we assume

1) the initial poses of all the vehicles are known since AUVs start the mission on the ocean surface, where global navigation satellite system signals are accessible and

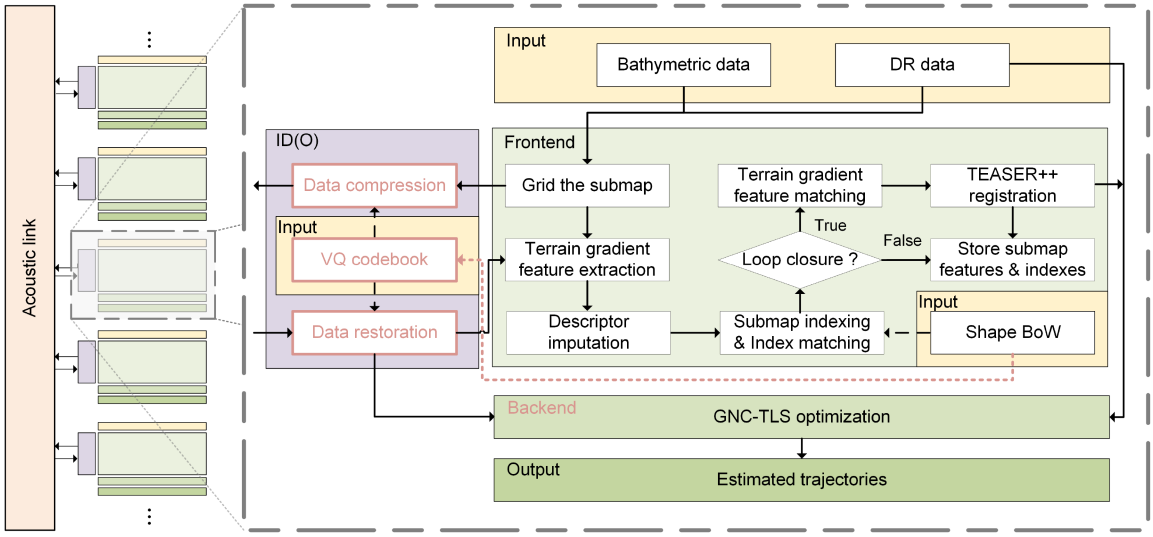

Fig. 6. Integration of ID(O) within TTT CSLAM framework. The blocks different from TTT SLAM with Shape BoW are marked in red. The red dotted line connecting the VQ codebook and Shape BoW represents that the former is constructed using the latter information. In this example, each vehicle maintains a SLAM solution.

2) errors in pitch, roll, and depth motions can be bounded by accurate measurements and the vehicle's inherent self-balancing capabilities.

During the mission, the poses of each vehicle are updated using a vehicle motion model

$$\mathbf{x}_t = \mathbf{x}_{t-1} + R(c_t + w) \tag{19}$$

where $\mathbf{x}_t = \begin{bmatrix} x_t & y_t & \psi_t \end{bmatrix}^{\mathrm{T}}$ is the vehicle's 2-D pose, R is a rotation matrix, c_t is the control at time t, and $w \sim N(0, Q)$ is Gaussian noise with covariance Q.

A bathymetric submap is constructed once a sufficient number of MBES pings are collected. After gridding the submap using inverse distance weighting interpolation, it is compressed using ID(O) and transmitted to the server vehicle (or broadcast to other vehicles) via acoustic communication.

When a submap is constructed by the vehicle itself or received from other vehicles, terrain gradient features are extracted, and the map is indexed using Shape BoW. Loop closure detection is performed by index matching. Subsequently, feature matching is conducted between this submap and the submaps with top-N highest index matching scores, and the results are provided to TEASER++ for registration. If the MSE between the registered map and the target map is below a threshold, a loop closure edge is added to the pose graph. Details of this procedure can be found in [17].

The pose graph structure of TTT CSLAM is demonstrated in Fig. 7. In addition to updating the nodes of the vehicle itself, when a submap is received from another vehicle, the corresponding pose node is added and connected to its preceding node of that vehicle in the graph. If a loop closure edge is added, the graph is optimized using GNC-TLS robust backend, and the trajectories are updated accordingly.

The output map is generated by merging submaps from all vehicles. In this process, these submaps are treated as point clouds rather than structured grid maps, enabling more flexible integration. The resulting joint point cloud can then be regridded to construct a digital elevation model for other missions, such as path planning or terrain analysis.

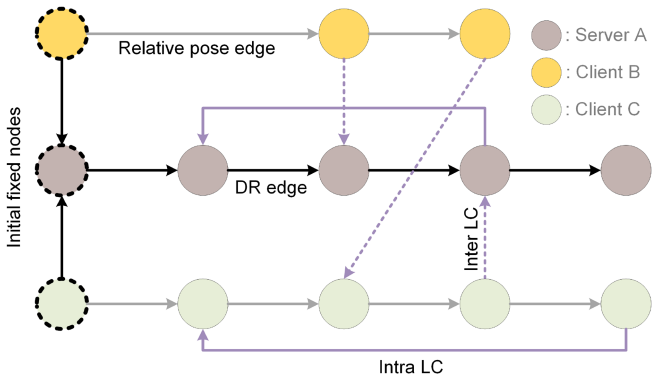

Fig. 7. Example of the pose graph structure of TTT CSLAM (in vehicle A). "Server" and "Client" refer to the server vehicle and client vehicles, respectively. "Inter LC" and "Intra LC" denote inter-robot and intra-robot loop closures, respectively. One submap sent from vehicle B to A is lost during transmission.

VIII. Experimental Results

Antarctica dataset [82] and Sheldrake seamount dataset [83], [84] were used to evaluate the performance of ID(O). These large-scale datasets were collected under diverse conditions, including differences in collection platforms, depths, covered areas, and terrain characteristics. All experiments were conducted in MATLAB and executed on a computer with an Intel Core i9-12900K CPU (5.20 GHz, single core) and 32 GB of RAM. The Shape BoW used in the experiments was constructed using the terrain gradient features extracted from Ripples and Reef datasets (see [17]) for generalization ability demonstration.

TABLE I
COMPRESSION METHODS AND THEIR PARAMETER SETTINGS

Method	Parameter name	Range
PNG[1]	-	-
JPEG	Image quality[2]	0%: 10%: 90%
JPEG 2K	CR[2]	1: 4: 40
SGP	Inducing input set size[2]	10: 10: 100
Quad[3]	Min-max threshold	1: 3: 30
PCA	No. of the transmitted third coefficients[4]	0: 5: ALL[5]
ID(O)	No. of the transmitted orientations	0: 5: ALL[5]

[1] PNG: Lossless method.
[2] Parameters of the built-in functions in MATLAB.
[3] Quad: Data stored in a quad-tree structure.
[4] The top two coefficients of each submap block are transmitted as a default.
[5] ALL: All third coefficients (orientations) of the submap blocks are transmitted.

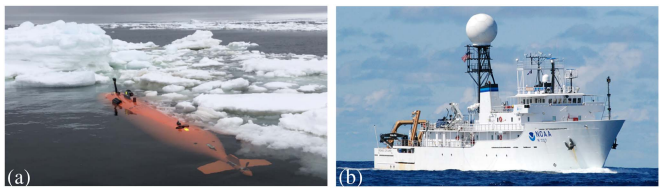

Fig. 8. Platforms for collecting Antarctica and Sheldrake seamount datasets. (a) Hugin 3000 AUV. (b) Okeanos Explorer. Information of Antarctica dataset can be found in [82], while that of Sheldrake seamount dataset is avaliable in [83] and [84].

A. Comparison of the Map Compression Methods

The first set of experiments evaluates the performance of ID(O) against the existing compression methods listed in Table I. We tested these methods on Antarctica dataset, which was collected beneath Thwaites Glacier using a Hugin 3000 AUV equipped with a Konsberg EM2040 MBES [see Fig. 8(a)] at an average depth of about 500 m and covers a region of about 10 km^2 [32], [82], [85].

The full dataset (the vehicle trajectory with corresponding MBES readings) was divided into 110 submaps. For each submap, we conducted map compression and restoration using different methods with various parameters. Since different settings of ID(O) lead to distinct behaviors, we named "IDO" as the method sending both $\mathbb{I}$, $\mathbb{D}$, and $\mathbb{O}$; "ID" as the method sending only $\mathbb{I}$ and $\mathbb{D}$; "ID(O)-N" as the method sending $\mathbb{I}$, $\mathbb{D}$, and the top-N essential main orientations.

The compression and restoration time, packet size, and the MSE between the input map and restored map were evaluated. In addition, we conducted feature matching between the terrain gradient features extracted from the restored maps and those from the input maps. The number of correctly matched features, feature preservation ratio (FPR), and feature mutation ratio (FMR) were calculated

$$\text{FPR} = \frac{\#\text{Matched}}{\#\text{Input}} \quad \text{and} \quad \text{FMR} = \frac{\#\text{Restored} - \#\text{Matched}}{\#\text{Restored}} \tag{20}$$

where #Input and #Restored are the number of the features extracted from the input map and that from the restored map, respectively, and #Matched is the number of the correctly matched features between these two maps. A higher FPR represents better preservation of valid features in the restored map, and a lower FMR indicates fewer invalid features introduced during restoration. Compared to MSE, which measures the restoration accuracy of general map shapes, FPR and FMR evaluate the ability of these compression methods to preserve fine map details.

The compression results on a single submap are demonstrated in Fig. 9 and Table II. For a fair comparison, we tuned the parameters of the baseline methods to ensure their CR were close to that of IDO in this experiment. Moreover, to minimize the impact of randomness, we present the average results over 100 test runs conducted on this submap in Table II. As a lossless method, PNG preserved nearly all map details but failed to achieve a high CR. In contrast, JPEG exhibited noticeable blocking artifacts, and JPEG 2K struggled to retain useful information at high CR.

The compression methods designed for bathymetric data delivered better results. Among these, IDO outperformed the other methods regarding restoration accuracy at high CR, achieving about 54% lower MSE, 41% higher FPR, and 51% lower FMR compared to the best baseline method using PCA. IDO also preserved the highest number of valid features among the lossy compression methods, which was 41% higher than the PCA method and sufficient for TEASER++ to perform submap registration.

In terms of compression time, the SGP method was the slowest due to the model training, while the other methods were efficient enough for online implementation. Regarding restoration time, the quad-tree-based method and PCA method enabled near-instantaneous restoration, while the other methods performed comparably (under 0.01 s).

Different parameters result in varying behaviors of the compression methods. Therefore, we adjusted their parameters and evaluated their performance. As shown in Fig. 10, the average results of the full dataset test are demonstrated by values changing with respect to CR [86]

$$\text{CR} = \frac{\text{Input map size}}{\text{Packet size}}. \tag{21}$$

As demonstrated in Fig. 10(a)–(d), the image-based methods failed to preserve details at high CR. In contrast, ID(O) outperformed PCA, the best baseline method, with a 49% lower MSE (0.63 versus 1.23), 31% higher FPR (0.67 versus 0.46), 48% lower FMR (0.23 versus 0.44), and 32% more matched features (1072 versus 730) at a CR of approximately 150.

Regarding computational time [see Fig. 10(e)], SGP requires model training, increasing the compression time proportional to the number of inducing points. Unlike the other compression methods that achieve near-instantaneous computation, ID(O) exhibits varying behaviors depending on its configuration. As illustrated in Fig. 10(e) and (f) and Table III, when all orientations were transmitted (IDO), it achieved fast compression (about 0.02 s) and restoration (about 0.004 s). When orientations were partially transmitted [ID(O)-N], NMSO optimization was required in both compression and restoration, resulting in a longer processing time. When orientations were excluded from sending (ID), the optimization was conducted only during restoration, while compression remained efficient (0.02 s).

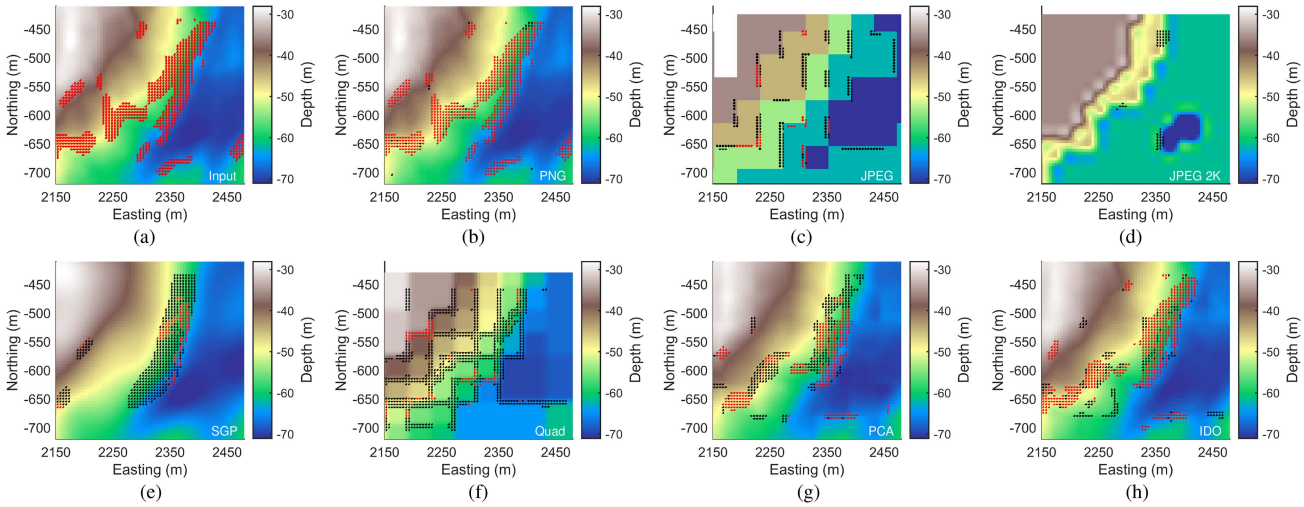

Fig. 9. One group of compression results on a single submap (after imputation). (a) Input map. (b)–(h) Restored maps by PNG, JPEG, JPEG 2K, SGP, Quad, PCA, and IDO, respectively. Matched features are marked in red, while the invalid features are indicated in black.

TABLE II
AVERAGE COMPRESSION RESULTS ON A SINGLE SUBMAP AFTER IMPUTATION

Type	Method	PS[1](byte)↓	MSE (m)↓	NMF[2]↑	FPR↑	FMR↓	CT[3](s)↓	RT[4](s)↓
Reference	PNG[5]	1763	0.01	1135	0.99	0.02	0.003	0.007
Compared	JPEG	432	10.75	54	0.05	0.92	0.002	0.003
	JPEG 2K	216	26.90	0	0	1	0.003	0.003
	SGP	191	1.11	262	0.23	0.57	2.49	0.005
	Quad	187	2.98	233	0.20	0.83	0.002	**0.001**
	PCA	**177**	0.35	499	0.44	0.45	**0.001**	**0.001**
	IDO	185	**0.16**	**846**	**0.74**	**0.22**	0.04	0.005

[1] PS: Packet size.
[2] NMF: The number of matched features.
[3] CT: Compression time.
[4] RT: Restoration time.
[5] PNG: Lossless method (for reference).
[6] Bold: Best-in-class (lossy methods).
[7] ↑: The higher the better.
[8] ↓: The lower the better.

TABLE III
AVERAGE COMPRESSION RESULTS OF PCA AND ID(O) ON THE FULL DATASET TEST

No. of 3rd elements	Method	PS[1](byte)↓	MSE (m)↓	NMF[2]↑	FPR↑	FMR↓	CT[3](s)↓	RT[4](s)↓	MEOE[5](deg)↓
0	PCA	**121**	5.24	224.52	0.16	0.84	**0.004**	**0.0007**	-
	ID	125	**2.14**	**878.31**	**0.56**	**0.39**	0.02	1.30	12.97
10	PCA	**131**	2.08	588.66	0.39	0.57	**0.004**	**0.0004**	-
	ID(O)-10	139	**0.78**	**1028.84**	**0.65**	**0.26**	1.30	1.32	5.17
20	PCA	**141**	1.36	711.43	0.45	0.46	**0.004**	**0.0004**	-
	ID(O)-20	149	**0.63**	**1071.55**	**0.67**	**0.23**	1.28	1.32	0.67
30	PCA	**150**	1.18	732.49	0.47	0.43	**0.004**	**0.0004**	-
	ID(O)-30	158	**0.58**	**1108.53**	**0.70**	**0.21**	1.18	1.21	0.40
ALL	PCA	**157**	1.16	736.90	0.47	0.42	**0.004**	**0.0004**	-
	IDO	165	**0.56**	**1121.42**	**0.71**	**0.20**	0.02	0.004	-

[1] PS: Packet size.
[2] NMF: The number of matched features.
[3] CT: Compression time.
[4] RT: Restoration time.
[5] MEOE: Median estimated orientation error.
[6] Bold: Best-in-class.
[7] ↑: The higher the better.
[8] ↓: The lower the better.

In the experiment, the parameters of PCA are the number of the transmitted third coefficients of the blocks, similar to that of the transmitted essential main orientations in ID(O). Therefore, we evaluated the performance of these two methods under the same number of the transmitted third elements.

As shown in Table III, all these two methods effectively reduced the data volume. When all the third elements of the blocks were transmitted, the average packet size was reduced by about 90% for both methods compared to PNG (1659 bytes). With the same number of transmitted third elements, ID(O) demonstrated an average of 55% lower MSE, 40% higher FPR, 52% lower FMR, and 41% more matched features than PCA, with a slightly larger packet size (4–8 bytes) due to the additional transmission of scaling factors.

Increasing the number of transmitted elements improves the restoration accuracy for both methods. However, despite a large improvement from ID to ID(O)-10, no notable accuracy gains were observed when transmitting more than 15 essential main orientations. Regarding the computational time, although ID(O) requires more processing time than PCA (especially when

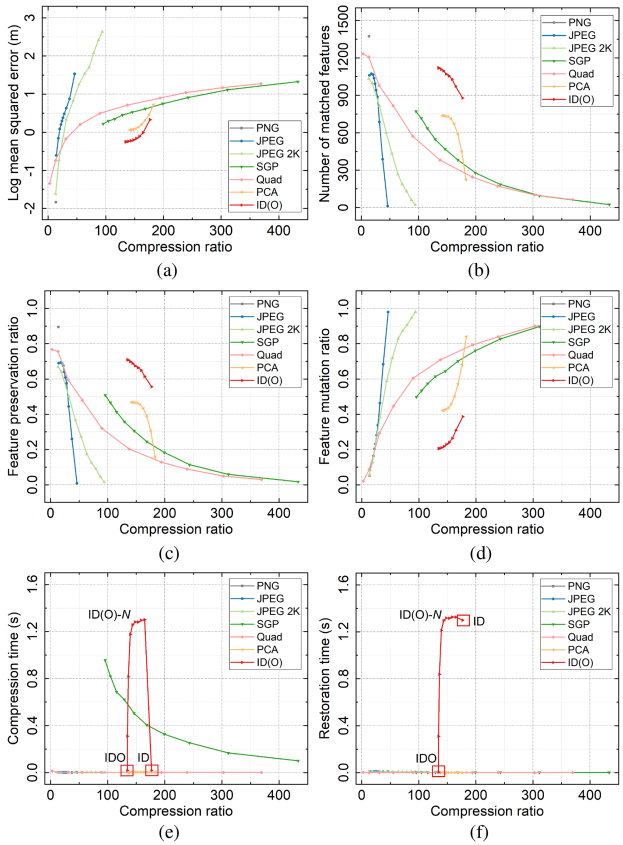

Fig. 10. Compression results using the methods with different parameters. Each sample represents an average result of a compression method using the same parameter on the full dataset (110 submaps) test. (a) Log MSE. (b) Number of matched features. (c) FPR. (d) FMR. (e) Compression time. (f) Restoration time.

requiring orientation estimation), it is a reasonable tradeoff considering the improved restoration accuracy.

As the number of transmitted orientations increases, the dimensionality of the problem (17) decreases, resulting in more accurate orientation estimation. When all the orientations were estimated (ID), the mean median estimated orientation error was 12.97°, which significantly dropped to 5.17° for ID(O)-10 and further to only 0.40° for ID(O)-30. Notably, the selection of the essential main orientations is designed to reduce the MSE between the input and restored maps rather than minimizing orientation estimation errors. For instance, a flat region with a large orientation error has minimal impact on the quality of the restored map.

Overall, the image-based methods performed poorly on this dataset, while the PCA-based method delivered the best results among the existing methods. At the same level of CR, ID(O) outperformed PCA, achieving 40% –55% better restoration accuracy in terms of MSE, FPR, FMR, and the number of matched features, while maintaining acceptable computational efficiency.

B. Case Study 1: TTT CSLAM on Antarctica Dataset

As discussed in Section VIII-A, SGP, PCA, and ID(O) significantly outperformed the other methods. Therefore, we integrated these methods with TTT CSLAM and tested them on Antarctica dataset under varying packet loss and substantial DR noise. In the experiment, the SGP method was adjusted to achieve a CR comparable to IDO, and the PCA methods using the top two and three coefficients (denoted as "PCA-S" and "PCA-F," corresponding to ID and IDO, respectively) were also evaluated.

TABLE IV
ONE GROUP OF TTT CSLAM RESULTS WITH DIFFERENT COMPRESSION METHODS ON ANTARCTICA DATASET

Type	Method	Consistency error (m) ↓	CT[1](s) ↓
Reference	DR	18.54	-
	PNG[2]	6.57	37.90
Compared	SGP[2]	18.75	151.19
	PCA-S[2]	17.83	**34.68**
	PCA-F[2]	9.70	41.19
	ID[2]	7.28	252.86
	IDO[2]	**6.97**	40.86

[1] CT: Computational time.
[2] TTT CSLAM with the compression method.
[3] ↓: The lower the better.
[4] Bold: Best-in-class.

As shown in Fig. 11(d), the trajectory of the vehicle and MBES readings were divided into two nonoverlapping parts (temporal perspective) at the segmentation point, representing the data from two vehicles [marked as "A" (Client) and "B" (Server)], respectively. These vehicles started mapping simultaneously, and for each AUV, the data was sequentially processed, with the next measurement being handled once the computation regarding the previous ping was completed. In this study, vehicle B also transmitted its collected maps to vehicle A to investigate the influence of map compression on computational time.

Moreover, to evaluate the influence of large DR noise on CSLAM, we additionally added Gaussian noise $e \sim N(0, Q)$ to the raw DR of each vehicle

$$DR_t = DR_{t-1} + R(c_t + e) \tag{22}$$

where DR_t, R, and c_t are the 2-D vehicle pose from DR, the rotation matrix, and the control input at time t, respectively. The covariance matrix is computed as $Q = L \times \text{diag}(1, 1, 0.1)$, where L represents the DR noise level. In this experiment, we generated ten sets of the DR data when $L = 0.05$ as the input to CSLAM, which is a challenging setting since the featureless SLAM methods with local registration failed to achieve satisfactory results due to large drift errors (see [17]).

In addition, we simulated map transmission failure by changing the percentage of the packet loss between 0% and 80%. These lost maps were randomly selected before each test but remained consistent across the CSLAM using different compression methods. The limitation of data rates was not considered for a fast execution (Refer to Section VIII-C for TTT CSLAM within an acoustic communication network).

One group of the test results is demonstrated in Figs. 11 and 12, and Table IV, where the DR noise level was 0.05 with no packet loss. Due to the lack of ground truth, we used the

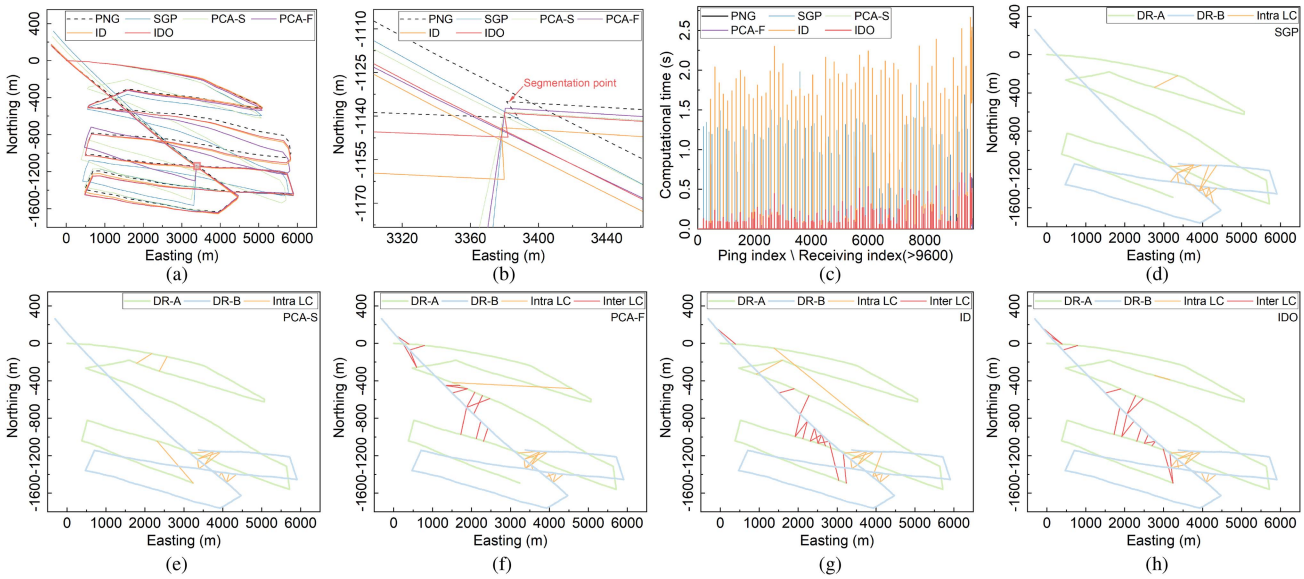

Fig. 11. One group of TTT CSLAM results with different compression methods on Antarctica dataset. (a) CSLAM trajectories. (b) Trajectories at the segmentation point. (c) Computational time (executed on MATLAB). After 9600 pings, vehicle B finished its mission and only received the maps from vehicle A. (d)–(h) Detected loop closures by the CSLAM with SGP, PCA-S, PCA-F, ID, and IDO, respectively (before the input to GNC-TLS backend). DR-A and DR-B represent the DR trajectories of vehicle A and B, respectively. Intra and Inter LC represent the intra-robot and inter-robot loop closures, respectively.

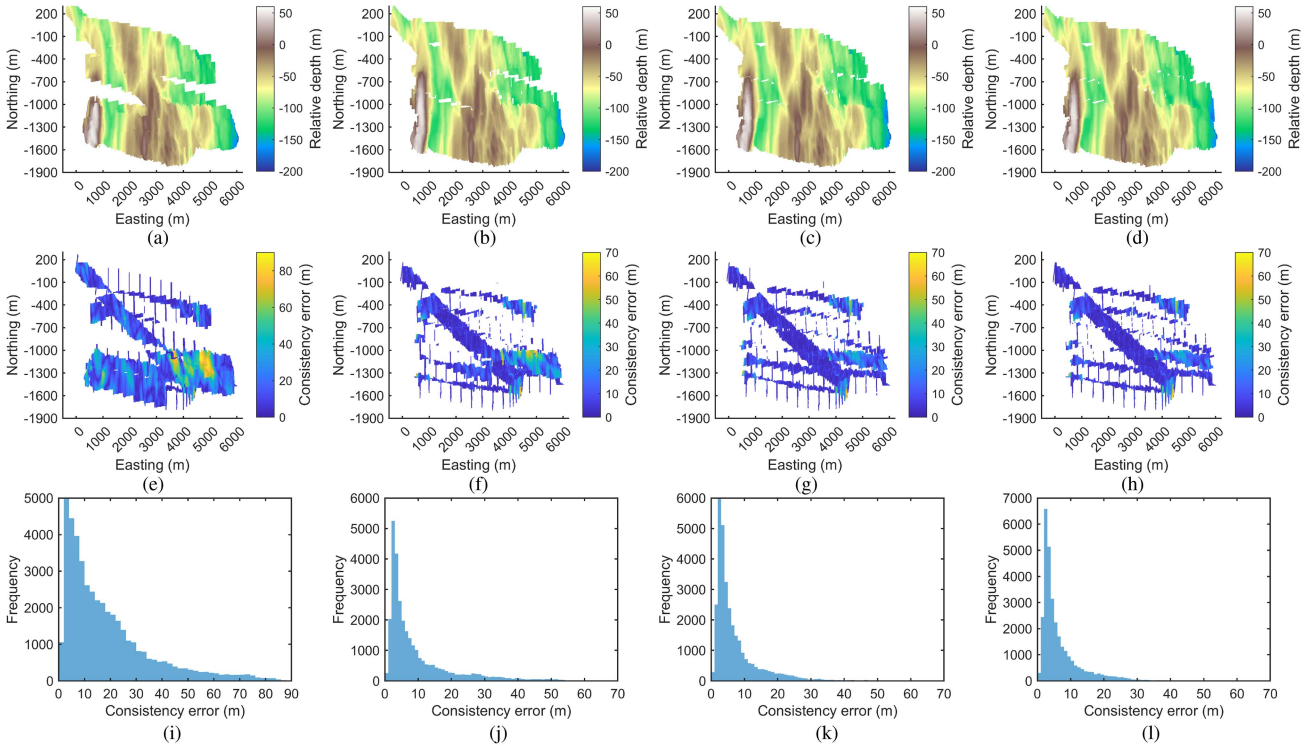

Fig. 12. One group of TTT CSLAM results with different compression methods on Antarctica dataset. (a)–(d) Constructed maps by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively. (e)–(h) Consistency error maps by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively. (i)–(l) Consistency error histograms by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively.

consistency error proposed in [87] to evaluate the CSLAM performance. As shown in Table IV, at the end of the mission, TTT CSLAM with PNG achieved the best performance, representing the theoretical best results that can be achieved by TTT CSLAM. In terms of accuracy, the CSLAM with ID and IDO outperformed that with PCA-F (best baseline method) by 24.95% and 28.14%, respectively, while TTT CSLAM with PCA-S and SGP failed to provide effective results due to the poor restored map quality at high CR.

Unlike other sensors, such as LiDAR and cameras, MBES measurements lack redundant information between consecutive scans, causing no data association can be conducted at the segmentation point. Therefore, the inconsistency of the trajectories at this point can effectively reflect CSLAM performance. As shown in Fig. 11(b), TTT CSLAM with PNG achieved the best results, followed by that with IDO and ID, all demonstrating a small inconsistency. In contrast, TTT CSLAM with PCA-based methods and SGP failed to connect the trajectories.

As demonstrated in Fig. 11(c), the execution time per ping of TTT CSLAM with PNG, PCA-based methods, and IDO was generally under 0.5 s. In contrast, when processing submaps, TTT CSLAM with SGP and ID required about 1.5 and 1.9 s due to the GPR model training in compression and NMSO optimization in restoration, respectively. However, as highlighted in Section VIII-A, ID achieved about 20% smaller packet size compared to PCA-F without substantially degrading map details, making it a viable option under strict communication constraints.

As illustrated in Fig. 11(d)–(h), the number of intra-robot loop closures detected by TTT CSLAM with different compression methods was all about 16 pairs. However, due to the loss of map details, no inter-robot loop closure was detected by the CSLAM with SGP and PCA-S. In comparison, TTT CSLAM with PNG, PCA-F, ID, and IDO detected 27, 13, 14, and 15 pairs of inter-robot loop closures, respectively. The validation of these loop closures cannot be conducted due to the lack of ground truth. However, it can be predicted that TTT CSLAM with IDO preserved more valid loop closures than that with other lossy methods because: 1) The graphs by the CSLAM with PCA-S, PCA-F, and ID clearly contained a few invalid intra-robot loop closures, which could potentially influence CSLAM performance despite the use of a robust backend. 2) ID(O) significantly outperformed the other methods in compression tests conducted on this dataset (see Section VIII-A). 3) From mapping results, TTT CSLAM with IDO achieved better accuracy.

The constructed maps, corresponding consistency error maps and the error histograms of TTT CSLAM with SGP, PCA-F, ID, and IDO are shown in Fig. 12. Totally 39.90%, 70.74%, 79.59%, and 80.59% of the errors were below 10 m, respectively. TTT CSLAM with ID and IDO achieved a comparable mapping quality to the CSLAM with PNG (83.77%). As shown in Fig. 12(e)–(h), the maps constructed by TTT CSLAM with SGP and PCA-F exhibited higher consistency errors, while those by the CSLAM with ID and IDO achieved lower errors, particularly within the region around the segmentation point.

The CSLAM results with different compression methods under substantial DR noise and varying levels of packet loss

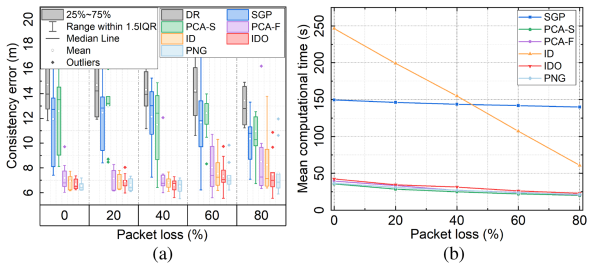

Fig. 13. Influence of packet loss on TTT CSLAM with different compression methods on Antarctica dataset (DR noise level: 0.05). (a) Consistency error. (b) Computational time (executed on MATLAB).

TABLE V
AVERAGE RESULTS OF TTT CSLAM WITH ID(O) UNDER UNETSTACK

Method	Consistency error (m)↓	CT[1](s)↓	RTR[2](%)↑
DR	14.12	-	-
ID[3]	7.24	12525	99.20
IDO[3]	6.80	12544	100

[1] CT: Computational time.
[2] RTR: Real-time ratio (Percentage of pings processed in under 1 s).
[3] TTT CSLAM with the compression method.
[4] ↑: The higher the better.
[5] ↓: The lower the better.

are demonstrated in Fig. 13 . Overall, TTT CSLAM with PNG, PCA-F, ID, and IDO outperformed that with the other methods in terms of accuracy. When the map loss was 20%, the mean consistency error of TTT CSLAM with SGP, PCA-S, PCA-F, ID, IDO, and PNG was 12.45, 13.00, 7.14, 6.90, 6.89, and 6.63 m, respectively. TTT CSLAM with ID and IDO achieved performance comparable to that with PNG. When the map loss increased to 60%, TTT CSLAM with PCA-F failed to maintain accuracy (mean error: 8.01 m) and stability (standard deviation: 1.90 m). In contrast, the mean and standard deviation of the consistency error of the CSLAM with ID were 7.53 and 1.33 m, respectively, and those of the CSLAM with IDO were 7.33 and 1.23 m, respectively, closely matching the performance of the CSLAM with PNG (7.26 and 1.11 m, respectively).

Under extreme cases (map loss: 80%), the performance of all methods declined significantly due to the lack of valid loop closures. However, the consistency error of TTT CSLAM with IDO remained comparable to that with PNG (all about 7.50 m), demonstrating the ability of IDO to preserve map details for effective data association.

Interestingly, the notable differences in the performance of PCA-F, ID(O), and PNG (see Tables II and III) did not dramatically influence the CSLAM results. This can be attributed to the strong robustness of TEASER++, which can provide valid registration results once the minimum requirements of the number of valid features and FPR rate are satisfied (higher than 200 and 0.40, respectively). Moreover, GNC-TLS robust backend further rejects invalid loop closures.

As illustrated in Fig. 13(b), TTT CSLAM with SGP and ID required more computational time due to model training and NMSO optimization respectively, while the other methods demonstrated the same level of efficiency in the experiment. When the packet loss increased from 0% to 80%, the computational time of TTT CSLAM with ID decreased from 246 to 61 s because of the reduced frequency of map restoration.

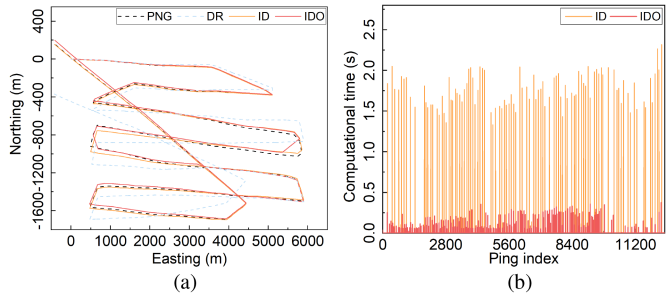

Fig. 14. One group of TTT CSLAM results with ID(O) on Antarctica dataset. (a) Trajectories generated by different navigation methods. PNG refers to TTT CSLAM with PNG method (no packet loss and Unetstack). (b) Computational time (executed on MATLAB).

In contrast, the CSLAM with SGP required about 145 s for execution regardless of packet loss since the computationally expensive model training is conducted during map compression. The computational time of TTT CSLAM with IDO decreased from 42 to 23 s, following a trend similar to the CSLAM with PNG (from 36 to 21 s).

Overall, TTT CSLAM with ID achieved a similar level of accuracy to the CSLAM with PCA while reducing the packet size by about 20%. TTT CSLAM with IDO closely matched that with PNG regarding accuracy and efficiency while reducing the packet size by approximately 90%.

C. Case Study 2: TTT CSLAM Under Unetstack

We also evaluated TTT CSLAM with ID(O) under realistic communication constraints on Antarctica dataset. Unetstack, an agent-based underwater network employed by Subnero acoustic modems, was used to simulate acoustic communication between vehicles. Details of the network can be found in [25].

In Unetstack, several key parameters influencing the performance of bathymetric CSLAM should be determined before tests, such as the distance between agents, packet loss rate, data rate, and sound speed. In the experiments, instead of online updating the vehicle positions in the network, the horizontal distance between the two nodes in Unetstack during the mission was fixed at 7000 m (with 500 m underwater), which is about the longest distance between two arbitrary poses in the raw vehicle trajectory, resulting in substantial communication latency. The probabilities of successfully detecting a packet and decoding a detected packet were both set to 0.8, with the data rate limited to only 200 bps and the sound speed configured at 1500 m/s.

The sensor readings were played back in real-time at 1 Hz. Once a map was collected by vehicle A (Client) and compressed using ID(O), the resulting packet was transmitted to vehicle B (Server). Upon reception, vehicle B performed data association and updated the graph. We conducted ten Monte Carlo tests when the DR noise level was 0.03.

One group of the results is demonstrated in Fig. 14, and the average results are summarized in Table V. TTT CSLAM with ID or IDO achieved performance comparable to the CSLAM with PNG under the acoustic communication network, demonstrating 48.73% and 51.84% better mapping accuracy compared to DR, respectively. As shown in Fig. 14(b), TTT CSLAM with IDO could be operated in real-time, with all computational time below 0.5 s. In comparison, the CSLAM with ID required more time for map restoration. However, this restoration process can be executed in parallel with CSLAM, making it also a practical method. As shown in Table V, due to the reduced time for transmitting small packets, TTT CSLAM with ID was about 20 s faster than that with IDO.

TABLE VI
AVERAGE RESULTS OF GA CSLAM WITH ID(O) ON ANTARCTICA DATASET

Type	Method	Consistency error (m) ↓	CT[1](s) ↓
Reference	DR	14.73	-
	PNG[2]	6.61	15712
Compared	ID[2]	6.97	15939
	IDO[2]	6.73	15736

[1] CT: Computational time.
[2] GA CSLAM with the compression method (no packet loss).
[3] ↓: The lower the better.

D. Extended Study: Application of ID(O) in Featureless Bathymetric CSLAM

The application of ID(O) is not limited to TTT CSLAM. In this study, we integrated it within a bathymetric CSLAM method using GA for data association (denoted as GA CSLAM) and tested the method on Antarctica dataset.

In GA CSLAM, potential overlapping submaps are identified by searching neighboring submaps around the vehicle position within an uncertainty ellipse derived from DR information. Submap registration is conducted by minimizing the MSE between the overlapping regions of the two maps using GA optimization through the adjustment of the 2-D vehicle pose (position and heading) [12]. The same as in [5], the crossover rate and mutation rate were adaptively adjusted based on the standard population diversity during optimization, and simulated binary crossover operators with polynomial mutation operators were implemented. To further enhance robustness, GNC-TLS backend was also used in GA CSLAM to reject invalid loop closures.

We conducted ten tests of GA CSLAM using 400 individuals to eliminate the influence of insufficient population size and randomness on the results. As demonstrated in Table VI, GA CSLAM with ID(O) showcased comparable mapping accuracy and efficiency to that with lossless compression, aligning with our findings in TTT CSLAM. However, since GA CSLAM selects all maps within the uncertainty ellipse for registration using computationally intensive GA, although it achieved satisfactory accuracy, its efficiency was degraded under a large population size and substantial DR drift errors.

E. Case Study 3: TTT CSLAM on Sheldrake Seamount Dataset

As shown in Figs. 8(b) and 15(a), the Sheldrake seamount dataset was collected using a Konsberg EM302 MBES mounted on Okeanos Explorer during the mission EX-14-04 L1 [83] and L3 [84] in 2014. The mountain is located about 5000 m underwater with a height of about 3000 m, and the missions covered a region of about 1000 km^2. As illustrated in Fig. 15(b), the trajectory of mission L1 follows a lawn-mowing path, while

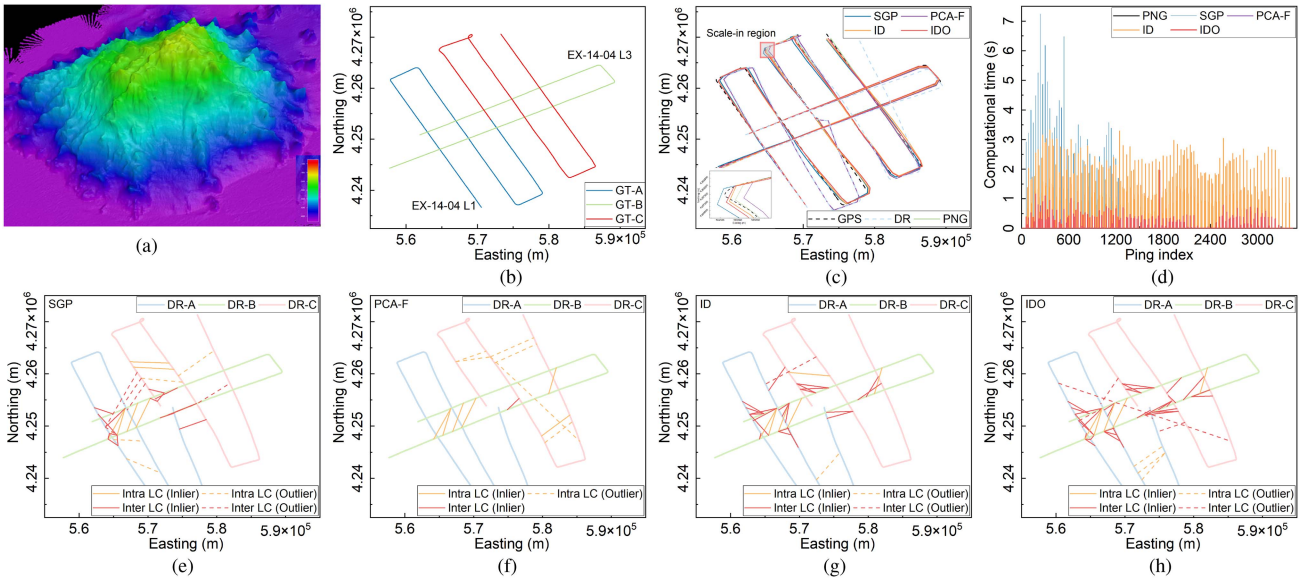

Fig. 15. One group of TTT CSLAM results with different compression methods on Sheldrake seamount dataset. (a) Sheldrake seamount [83], [84]. (b) Ground truth trajectories. The L1 trajectory was divided into two for CSLAM using three vehicles. GT-(A-C) represent the ground truth trajectories of vehicles A, B, and C, respectively. (c) CSLAM trajectories. (d) Computational time (executed on MATLAB). (e)-(h) Detected loop closures by the CSLAM with SGP, PCA-F, ID, and IDO, respectively (before the input to GNC-TLS backend). DR-(A-C) represent the DR trajectories of vehicles A, B, and C, respectively. Intra and Inter LC represent the intra-robot and inter-robot loop closures, respectively.

TABLE VII
ONE GROUP OF TTT CSLAM RESULTS WITH DIFFERENT COMPRESSION METHODS ON SHELDRAKE SEAMOUNT DATASET

Type	Method	Mean trajectory error (m)↓	Registration error (m)↓	CT[1](s)↓	IR[2]↑	IN[3]↑
Reference	DR	753.20	60.89	-	-	-
	PNG[4]	143.85	13.98	44.95	0.83	100
Compared	SGP[4]	229.29	24.85	162.78	0.69	22
	PCA-F[4]	443.34	45.55	**42.20**	0.54	7
	ID[4]	**152.59**	17.14	310.29	**0.94**	29
	IDO[4]	158.27	**15.20**	55.88	0.84	**41**

[1] CT: Computational time.
[2] IR: Inlier rate of the detected loop closures.
[3] IN: Inlier number of the detected loop closures.
[4] TTT CSLAM with the compression method.
[5] Bold: Best-in-class.
[6] ↑: The higher the better.
[7] ↓: The lower the better.

that of L3 across the L1 trajectory. The overlap between L1 and L3 trajectories makes the dataset well-suited for testing CSLAM.

However, the trajectory length of mission L1 is significantly longer than that of L3. Therefore, we divided the L1 trajectory (with measurements) into two nonoverlapping parts, representing the data collected using different vehicles (marked as A and C), while the data from mission L3 was assumed to be collected using vehicle B. In the experiment, each vehicle maintained a SLAM solution, and these vehicles started mapping simultaneously. For each AUV, the data was sequentially processed, with the next measurement being handled once the computation regarding the previous ping was completed. In this article, we present the CSLAM results from vehicle B to avoid redundancy.

We evaluated the influence of DR drift errors (the DR noise level changed from 0.01 to 0.05) and packet loss (changed from 0% to 80%) on the results. For each noise level, ten sets of DR data were generated as the input to the CSLAM with different compression methods. For fairness, the parameters of SGP were adjusted to keep its packet size close to that of IDO, which was approximately 20% larger than that of ID.

One group of the results when the DR noise level was 0.04 with no packet loss is demonstrated in Figs. 15 and 16, and Table VII. At the end of the mission, TTT CSLAM with ID and IDO achieved similar trajectory accuracy, comparable to the CSLAM with PNG and outperforming that with SGP (best baseline method) by about 30%. Although TTT CSLAM with ID exhibited a 5.68 m lower trajectory error compared to the CSLAM with IDO, its map registration error was about 11.32% higher due to the relatively low map restoration accuracy.

In terms of computational time, TTT CSLAM with IDO was slightly slower than the CSLAM with PNG, but about three times faster than that with SGP. In contrast, due to the increased number of blocks divided from large-scale submaps, the CSLAM with ID required more time for orientation estimation during restoration, resulting in slower execution.

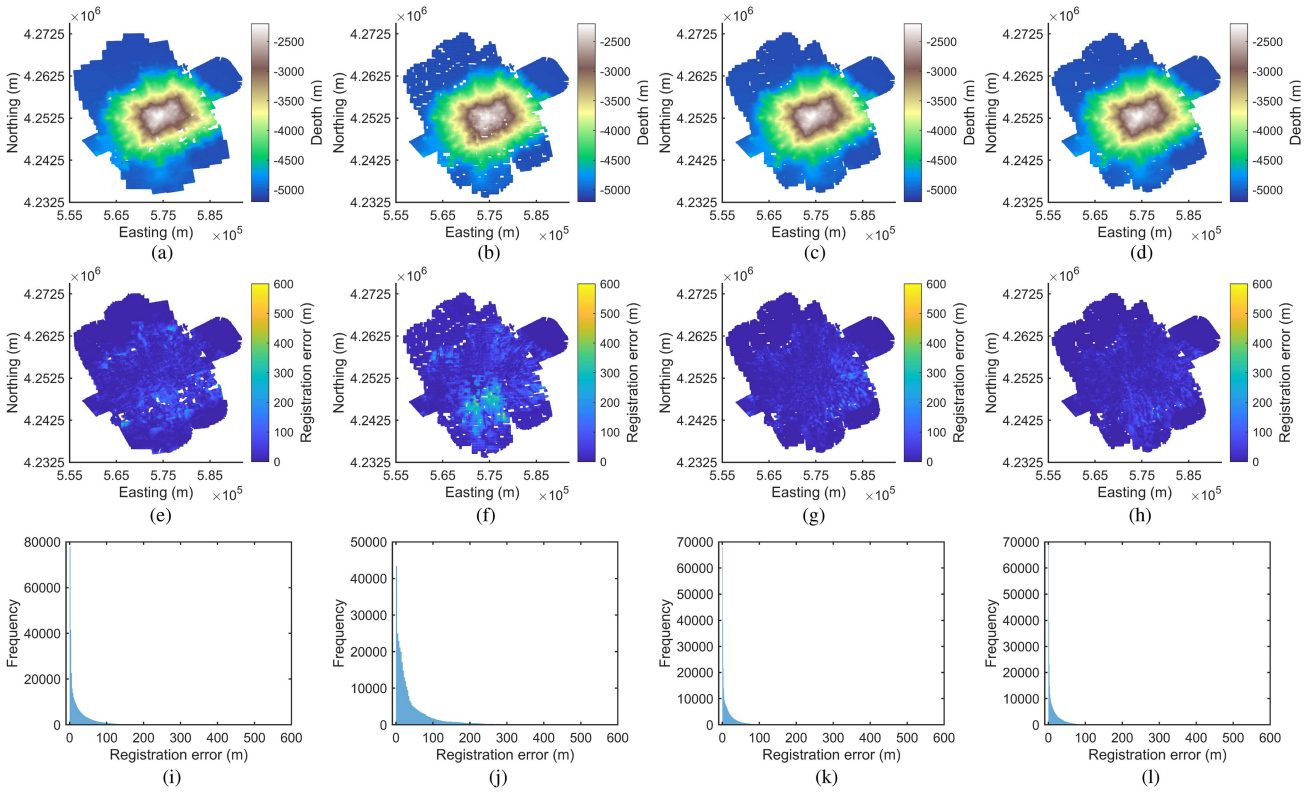

Fig. 16. One group of TTT CSLAM results with different compression methods on Sheldrake seamount dataset. (a)–(d) Constructed maps by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively. (e)–(h) Registration error maps by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively. (i)–(l) Registration error histograms by TTT CSLAM with SGP, PCA-F, ID, and IDO, respectively.

As shown in Fig. 15(e)–(h), we also evaluated the inlier rate and inlier number of the loop closures detected by different methods. TTT CSLAM with IDO identified the highest number of valid loop closures, which was 46.34% more than the CSLAM with SGP. In contrast, that with PCA-F detected only seven pairs of valid loop closures. TTT CSLAM with ID(O) methods also achieved inlier rates comparable to the CSLAM with PNG, which were significantly higher than that with SGP and PCA-F.

The constructed maps, registration error maps and error histograms by TTT CSLAM with SGP, PCA-F, ID, and IDO are demonstrated in Fig. 16. Total 72.35%, 59.17%, 81.89%, and 82.90% of the depth errors were below 30 m, respectively. TTT CSLAM with ID and IDO achieved mapping quality comparable to the CSLAM with PNG (84.86%) and exhibited smaller errors within the regions with significant depth changes compared to that with SGP and PCA-F.

Interestingly, compared to ID(O), which maintained consistent performance across different scenarios, PCA and SGP exhibited instability in these case studies. For PCA, the eigensurface database was trained using the Ripples and Reef datasets (like Shape BoW). However, due to the substantial differences between the training and test data, the top three PCA coefficients were insufficient for accurate map restoration, leading to unsatisfactory performance on Sheldrake seamount dataset. Regarding SGP, it preserved the general shapes of the maps when using a small set of inducing points, making it ideal for compressing large-scale coarse maps (e.g., 60 m resolution maps on Shelderake seamount dataset), but ineffective for preserving fine details of high-resolution maps (e.g., 5 m resolution maps on Antarctica dataset).

The CSLAM results with different compression methods under varying levels of DR noise and packet loss are shown in Figs. 17 and 18. We demonstrated map registration errors instead of trajectory errors since the CSLAM mapping accuracy can reflect both positioning accuracy and map restoration accuracy. Overall, TTT CSLAM with PCA-F failed to provide effective results on this dataset due to the generalization problem. Regarding the other methods, an increase in packet loss leads to higher mapping errors. As shown in Fig. 17(a), when the DR noise level was 0.02 with 20% packet loss, the mean map registration errors of TTT CSLAM with SGP, ID, IDO, and PNG were 27.39, 24.85, 20.09, and 14.14 m, respectively, which increased to 35.82, 33.25, 24.67, and 18.63 m, respectively, when packet loss was 60%. Similarly, higher DR noise causes greater mapping errors. With a DR noise level of 0.01 and 40% packet loss, the registration errors of these methods were 24.13, 22.65, 15.25, and 12.40 m, respectively. These errors increased to 43.65, 33.96, 30.31, and 21.22 m, respectively, when the DR noise level changed to 0.05. TTT CSLAM with ID demonstrated a comparable level of accuracy to the CSLAM with SGP, while that with IDO outperformed that with SGP by about 31%.

As demonstrated in Fig. 17(b), compared to TTT CSLAM with PNG, the number of the inlier loop closures detected by the CSLAM with lossy compression methods decreased significantly. When the DR noise level was 0.02 with 20% packet loss,

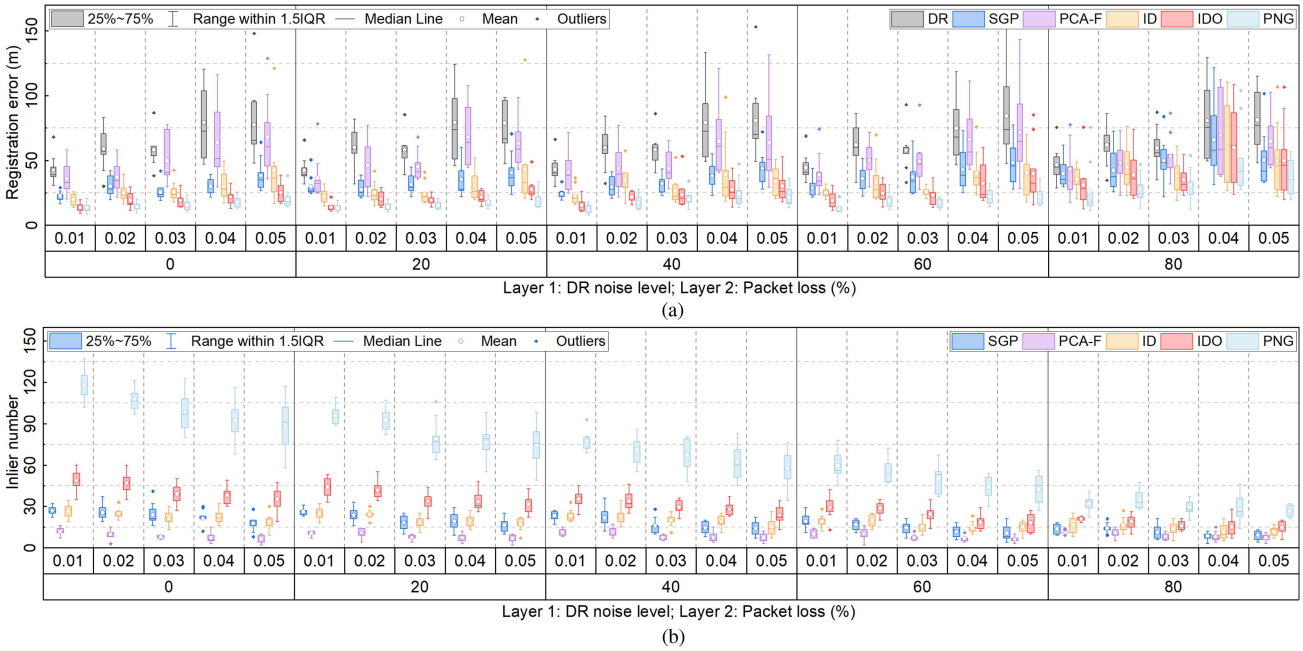

Fig. 17. Influence of DR noise and packet loss on TTT CSLAM with different compression methods on Sheldrake seamount dataset. (a) Map registration error. (b) Inlier number of the detected loop closures (before the input to GNC-TLS backend).

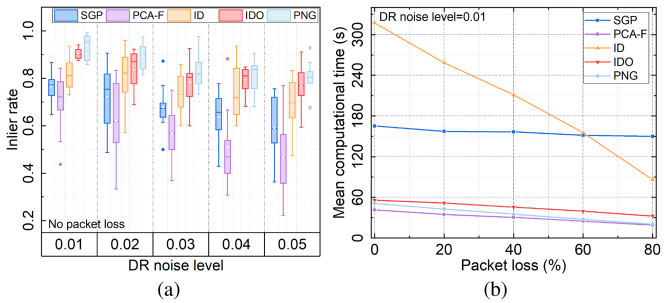

Fig. 18. Influence of DR noise or packet loss on TTT CSLAM with different compression methods on Sheldrake seamount dataset. (a) Inlier rate of the detected loop closures (before the input to GNC-TLS backend). (b) Computational time (executed on MATLAB).

the detected inlier loop closures by TTT CSLAM with SGP, ID, IDO, and PNG were 23.6, 24.0, 41.6, and 92.7 pairs, respectively. Increasing DR noise or packet loss further reduces the number of detected loop closures. When the DR noise level was 0.05 with 40% packet loss, the number of detected loop closures by TTT CSLAM with these methods dropped to 14.0, 15.1, 23.7, and 57.3 pairs, respectively. TTT CSLAM with ID detected a similar number of valid loop closures as the CSLAM with SGP, while that with IDO detected about 40% more loop closures.

As illustrated in Fig. 18(a), since packet loss has a minor effect on the inlier rates of detected loop closures, we demonstrate the influence of DR noise on inlier rates. When the DR noise level was 0.01 with no packet loss, the inlier rates of TTT CSLAM with different methods were 0.76 (SGP), 0.69 (PCA-F), 0.82 (ID), 0.90 (IDO), and 0.94 (PNG), which decreased to 0.59, 0.47, 0.69, 0.77, and 0.80, respectively, when the DR noise level increased to 0.05. The inlier rates of TTT CSLAM with ID were slightly higher than that with SGP, while the CSLAM with IDO achieved inlier rates comparable to that with PNG.

As shown in Fig. 18(b), because computational time was only slightly affected by DR noise, we evaluated the impact of packet loss on the computational time of the CSLAM. Similar to the trend observed in Case Study 1, the computational time of TTT CSLAM with ID decreased significantly with the increasing packet loss due to the reduced frequency of map restoration. Specifically, when packet loss increased from 0% to 80%, the computational time of TTT CSLAM with ID dropped from approximately 318 s to about 86 s. In contrast, TTT CSLAM with SGP required approximately 150 s for execution regardless of packet loss. TTT CSLAM with PCA-F, IDO, and PNG exhibited similar efficiency, requiring about 50 s with no packet loss and 25 s with 80% packet loss, which were at least about 66.7% faster than the CSLAM with SGP.

IX. Conclusion

In this work, we introduce a novel mapping data quantization method called ID(O) and integrate it with a feature-based bathymetric CSLAM framework named TTT CSLAM. Utilizing the general patterns captured from the local seabed terrain changes, ID(O) enables efficient mapping data transmission in bathymetric CSLAM. Based on the experimental results, we draw the following conclusions.

1) At the same level of CR, ID(O) achieves approximately 40% higher map restoration accuracy than the PCA-based method while maintaining acceptable computational efficiency.

2) TTT CSLAM with IDO closely matches that with lossless compression regarding mapping accuracy and efficiency while sending significantly smaller data packets.
3) TTT CSLAM with ID is comparable to that with PCA or SGP in terms of accuracy while reducing the packet size by about 20%.
4) TTT CSLAM with ID(O) demonstrates robustness against large DR drift errors and substantial packet loss across diverse environments, and it can be effectively implemented under an acoustic communication network in challenging settings (200 bps data rates, about 40% packet loss, and high latency).

Although ID(O) has showcased promising performance across diverse environments, its codebook is fixed before the mission, which may lead to increased restoration errors when the terrain shapes are significantly different from those stored in the codebook. Therefore, the online adaptation of the codebook to new environments by dynamically adjusting scaling factors based on map restoration errors can further improve the generalization ability of ID(O). Moreover, the extrapolation errors introduced during the orientation estimation can influence the accuracy of ID. The estimation also becomes computationally expensive when restoring large-scale maps. Future work will focus on addressing these limitations. Additionally, the extension of TTT CSLAM by integrating a communication-efficient collaborative backend (e.g., [88]) will also be investigated.

ACKNOWLEDGMENT

The authors would like to thank the editors and reviewers for their insightful comments. The authors would also like to thank Dr. Dongha Chung for the constructive feedback on this article.

REFERENCES

[1] A. G. C. Graham et al., "Rapid retreat of Thwaites glacier in the pre-satellite era," *Nature Geosci.*, vol. 15, no. 9, pp. 706–713, Sep. 2022.
[2] I. Torroba, N. Bore, and J. Folkesson, "Towards autonomous industrial-scale bathymetric surveying," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6377–6382.
[3] J. Melo and A. Matos, "Survey on advances on terrain based navigation for autonomous underwater vehicles," *Ocean Eng.*, vol. 139, pp. 250–264, Jul. 2017.
[4] X.-F. Liu, Y. Fang, Z.-H. Zhan, Y.-L. Jiang, and J. Zhang, "A cooperative computation algorithm for dynamic multiobjective multi-AUV path planning," *IEEE Trans. Industr. Informat.*, vol. 20, no. 1, pp. 669–680, Jan. 2024.
[5] C. Qi, T. Ma, Y. Li, L. Lv, and Y. Ling, "An efficient loop closure detection method for communication-constrained bathymetric cooperative SLAM," *Ocean Eng.*, vol. 304, 2024, Art. no. 117720.
[6] L. Paull, G. Huang, M. Seto, and J. J. Leonard, "Communication-constrained multi-AUV cooperative SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 509–516.
[7] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, online, and outlier resilient SLAM for robotic teams," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1656–1663, Apr. 2020.
[8] D. Van Opdenbosch and E. Steinbach, "Collaborative visual SLAM using compressed feature exchange," *IEEE Robot. Autom. Lett.*, vol. 4, no. 1, pp. 57–64, Jan. 2019.
[9] Y. Xie et al., "RDC-SLAM: A real-time distributed cooperative SLAM system based on 3D LiDAR," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14721–14730, Sep. 2022.
[10] Y. Tian, Y. Chang, F. Herrera Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-multi: Robust, distributed, dense metric-semantic SLAM for multi-robot systems," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. Aug. 2022–2038, 2022.
[11] M. Y. I. Zia, J. Poncela, and P. Otero, "State-of-the-art underwater acoustic communication modems: Classifications, analyses and design challenges," *Wirel. Pers. Commun.*, vol. 116, no. 2, pp. 1325–1360, 2021.
[12] T. Ma et al., "Communication-constrained cooperative bathymetric simultaneous localisation and mapping with efficient bathymetric data transmission method," *J. Navig.*, vol. 75, no. 4, pp. 1000–1016, 2022.
[13] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 5220–5227.
[14] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "Bathymetric particle filter SLAM using trajectory maps," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1409–1430, 2012.
[15] M. Teng, L. Ye, Z. Yuxin, J. Yanqing, Z. Qianyi, and A. M. Pascoal, "Efficient bathymetric SLAM with invalid loop closure identification," *IEEE-ASME Trans. Mechatron.*, vol. 26, no. 5, pp. 2570–2580, Oct. 2021.
[16] Q. Zhang and J. Kim, "TTT SLAM: A feature-based bathymetric SLAM framework," *Ocean Eng.*, vol. 294, 2024, Art. no. 116777.
[17] Q. Zhang and J. Kim, "Shape BoW: Generalized bag of words for appearance-based loop closure detection in bathymetric SLAM," *IEEE Robot. Autom. Lett.*, vol. 9, no. 9, pp. 7405–7412, Sep. 2024.
[18] G. Scarmana, "Lossless data compression of grid-based digital elevation models: A PNG image format evaluation," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. II-5, pp. 313–319, 2014.
[19] M. Platings and A. M. Day, "Compression of large-scale terrain data for real-time visualization using a tiled quad tree," *Comput. Graph. Forum*, vol. 23, no. 4, pp. 741–759, 2004.
[20] C. Murphy, R. Y. Wang, and H. Singh, "Seafloor image compression with large tilesize vector quantization," in *Proc. IEEE/OES Auton. Underwater Veh.*, 2010, pp. 1–8.
[21] A. Danckaers and M. L. Seto, "Transmission of images by unmanned underwater vehicles," *Auton. Robots*, vol. 44, no. 1, pp. 3–24, Jan. 2020.
[22] A. Vasuki and P. Vanathi, "A review of vector quantization techniques," *IEEE Potentials*, vol. 25, no. 4, pp. 39–47, Jul./Aug. 2006.
[23] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 314–333, Apr. 2021.
[24] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1127–1134, Apr. 2020.
[25] M. Chitre, R. Bhatnagar, and W.-S. Soh, "UnetStack: An agent-based software stack and simulator for underwater networks," in *Proc. OCEANS*, 2014, pp. 1–10.
[26] S. Rane and G. Sapiro, "Evaluation of JPEG-LS, the new lossless and controlled-lossy still image compression standard, for compression of high-resolution elevation data," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 10, pp. 2298–2306, Oct. 2001.
[27] G. Scarmana and K. McDougall, "A pyramid approach to lossless data compression of grid-based digital elevation models," in *Proc. IEEE Geosci. Remote Sens. Symp.*, 2014, pp. 2503–2506.
[28] W. Maleika and P. Forczmański, "Adaptive modeling and compression of bathymetric data with variable density," *IEEE J. Ocean. Eng.*, vol. 45, no. 4, pp. 1353–1369, Oct. 2020.
[29] P. Forczmański and W. Maleika, "Predicting the number of DCT coefficients in the process of seabed data compression," in *Proc. 16th Int. Conf. Comput. Anal. Images Patterns*, 2015, pp. 77–87.
[30] P. Forczmański and W. Maleika, "Near-lossless PCA-based compression of seabed surface with prediction," in *Proc. 12th Int. Conf. Image Anal. Recognit.*, 2015, pp. 119–128.
[31] N. Bore, I. Torroba, and J. Folkesson, "Sparse Gaussian process SLAM, storage and filtering for AUV multibeam bathymetry," in *Proc. IEEE/OES Auton. Underwater Veh. Workshop*, 2018, pp. 1–6.
[32] I. Torroba, C. I. Sprague, and J. Folkesson, "Fully-probabilistic terrain modelling and localization with stochastic variational Gaussian process maps," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 8729–8736, Oct. 2022.
[33] Z. Cong et al., "A storage-saving quadtree-based multibeam bathymetry map representation method," *J. Mar. Sci. Eng.*, vol. 11, no. 4, 2023.
[34] Y. Hu, W. Yang, Z. Ma, and J. Liu, "Learning end-to-end lossy image compression: A benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 8, pp. 4194–4211, Aug. 2022.

[35] J. Chen, Y. Fang, A. Khisti, A. Özgür, and N. Shlezinger, "Information compression in the AI era: Recent advances and future challenges," *IEEE J. Sel. Areas Commun.*, vol. 43, no. 7, pp. 2333–2348, Jul. 2025.

[36] R. Abbasi, A. K. Bashir, H. J. Alyamani, F. Amin, J. Doh, and J. Chen, "LiDAR point cloud compression, processing and learning for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 1, pp. 962–979, Jun. 2023.

[37] M. Wang, R. Huang, W. Xie, Z. Ma, and S. Ma, "Compression approaches for LiDAR point clouds and beyond: A survey," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 21, no. 7, 2025, Art. no. 188.

[38] N. Krishnaraj, M. Elhoseny, M. Thenmozhi, M. M. Selim, and K. Shankar, "Deep learning model for real-time image compression in Internet of Underwater Things (IoUT)," *J. Real-Time Image Process.*, vol. 17, no. 6, pp. 2097–2111, 2020.

[39] K. Anjum, Z. Li, and D. Pompili, "Acoustic channel-aware autoencoder-based compression for underwater image transmission," in *Proc. Underwater Commun. Netw. Conf.*, 2022, pp. 1–5.

[40] J. Liu, F. Yuan, C. Xue, Z. Jia, and E. Cheng, "An efficient and robust underwater image compression scheme based on autoencoder," *IEEE J. Ocean. Eng.*, vol. 48, no. 3, pp. 925–945, Jul. 2023.

[41] C. Gomes et al., "Lossy neural compression for geospatial analytics: A review," *IEEE Geosci. Remote Sens. Mag.*, vol. 13, no. 3, pp. 97–135, Sep. 2025.

[42] D. Zou and P. Tan, "CoSLAM: Collaborative visual SLAM in dynamic environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 354–366, Feb. 2013.

[43] P.-Y. Lajoie and G. Beltrame, "Swarm-SLAM: Sparse decentralized collaborative simultaneous localization and mapping framework for multi-robot systems," *IEEE Robot. Autom. Lett.*, vol. 9, no. 1, pp. 475–482, Jan. 2024.

[44] S. Fang and H. Li, "Multi-vehicle cooperative simultaneous LiDAR SLAM and object tracking in dynamic environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 9, pp. 11411–11421, Sep. 2024.

[45] Y. Chang et al., "LAMP 2.0: A robust multi-robot SLAM system for operation in challenging large-scale underground environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9175–9182, Oct. 2022.

[46] M. Karrer, P. Schmuck, and M. Chli, "CVI-SLAM—collaborative visual-inertial SLAM," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 2762–2769, Oct. 2018.

[47] X. Pan, G. Huang, Z. Zhang, J. Li, H. Bao, and G. Zhang, "Robust collaborative visual-inertial SLAM for mobile augmented reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 30, no. 11, pp. 7354–7363, Nov. 2024.

[48] P. Schmuck and M. Chli, "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *J. Field Robot.*, vol. 36, no. 4, pp. 763–781, 2019.

[49] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, "COVINS: Visual-inertial SLAM for centralized collaboration," in *Proc. IEEE Int. Symp. Mixed Augmented Reality Adjunct*, 2021, pp. 171–176.

[50] M. Patel, M. Karrer, P. Bänninger, and M. Chli, "COVINS-G: A generic back-end for collaborative visual-inertial SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 2076–2082.

[51] H. Xu, P. Liu, X. Chen, and S. Shen, "D^2SLAM: Decentralized and distributed collaborative visual-inertial SLAM system for aerial swarm," *IEEE Trans. Robot.*, vol. 40, pp. 3445–3464, 2024.

[52] I. Deutsch, M. Liu, and R. Siegwart, "A framework for multi-robot pose graph SLAM," in *Proc. IEEE Int. Conf. Real-time Comput. Robot.*, 2016, pp. 567–572.

[53] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit*, 2016, pp. 5297–5307.

[54] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4802–4809.

[55] T. Cieslewski and D. Scaramuzza, "Efficient decentralized visual place recognition using a distributed inverted index," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 640–647, Apr. 2017.

[56] Y. Tian, K. Khosoussi, and J. P. How, "A resource-aware approach to collaborative loop-closure detection with provable performance guarantees," *Int. J. Robot. Res.*, vol. 40, no. 10-11, pp. 1212–1233, 2021.

[57] A. Cunningham, M. Paluri, and F. Dellaert, "DDF-SAM: Fully distributed SLAM using constrained factor graphs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 3025–3030.

[58] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, "Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models," *Int. J. Robot. Res.*, vol. 36, no. 12, pp. 1286–1311, 2017.

[59] T. Fan and T. D. Murphey, "Majorization minimization methods for distributed pose graph optimization," *IEEE Trans. Robot.*, vol. 40, pp. 22–42, 2024.

[60] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, "Distributed certifiably correct pose-graph optimization," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 2137–2156, Dec. 2021.

[61] P.-Y. Lajoie, B. Ramtoula, F. Wu, and G. Beltrame, "Towards collaborative simultaneous localization and mapping: A survey of the current research landscape," *Field Robot.*, vol. 2, pp. 971–1000, 2022.

[62] J. Kim, "Cooperative localization and unknown currents estimation using multiple autonomous underwater vehicles," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2365–2371, Apr. 2020.

[63] M. Bai, Y. Huang, Y. Zhang, and F. Chen, "A novel heavy-tailed mixture distribution based robust Kalman filter for cooperative localization," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3671–3681, May 2021.

[64] Y. Huang, M. Bai, Y. Li, Y. Zhang, and J. Chambers, "An improved variational adaptive Kalman filter for cooperative localization," *IEEE Sens. J.*, vol. 21, no. 9, pp. 10775–10786, May 2021.

[65] Y. Li, Y. Wang, W. Yu, and X. Guan, "Multiple autonomous underwater vehicle cooperative localization in anchor-free environments," *IEEE J. Ocean. Eng.*, vol. 44, no. 4, pp. 895–911, Oct. 2019.

[66] A. Wiktor and S. Rock, "Collaborative multi-robot localization in natural terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 4529–4535.

[67] Y. T. Tan, M. Chitre, and F. S. Hover, "Cooperative bathymetry-based localization using low-cost autonomous underwater vehicles," *Auton. Robots*, vol. 40, no. 7, pp. 1187–1205, Oct. 2016.

[68] L. Jiang, Y. Li, W. Yu, and X. Guan, "Cooperative localization for asynchronous AUVs using time difference of communication in underwater anchor-free environments," *IEEE Trans. Cybern.*, vol. 54, no. 11, pp. 6531–6544, Nov. 2024.

[69] I. Skog and P. Handel, "Synchronization by two-way message exchanges: Cramér-Rao bounds, approximate maximum likelihood, and offshore submarine positioning," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2351–2362, Apr. 2010.

[70] W. Gao, Y. Liu, B. Xu, and Y. Che, "An improved cooperative localization method for multiple autonomous underwater vehicles based on acoustic round-trip ranging," in *Proc. IEEE/ION Position Location Navig. Symp.*, 2014, pp. 1420–1423.

[71] M. Pfingsthorn, A. Birk, and H. Bülow, "An efficient strategy for data exchange in multi-robot mapping under underwater communication constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 4886–4893.

[72] M. Pfingsthorn, A. Birk, N. Vaskevicius, and K. Pathak, "Cooperative 3D mapping under underwater communication constraints," in *Proc. MTS/IEEE OCEANS*, 2011, pp. 1–9.

[73] J. McConnell, Y. Huang, P. Szenher, I. Collado-Gonzalez, and B. Englot, "DRACo-SLAM: Distributed robust acoustic communication-efficient SLAM for imaging sonar equipped underwater robot teams," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 8457–8464.

[74] Y. Ling, Y. Li, T. Ma, Z. Cong, S. Xu, and Z. Li, "Active bathymetric SLAM for autonomous underwater exploration," *Appl. Ocean Res.*, vol. 130, Jan. 2023, Art. no. 103439.

[75] G. Wang, D. Garcia, Y. Liu, R. de Jeu, and A. J. Dolman, "A three-dimensional gap filling method for large geophysical datasets: Application to global satellite soil moisture observations," *Environ. Modell. Softw.*, vol. 30, pp. 139–142, 2012.

[76] L. Stripinis, J. Kůdela, and R. Paulavičius, "Benchmarking derivative-free global optimization algorithms under limited dimensions and large evaluation budgets," *IEEE Trans. Evol. Comput.*, vol. 29, no. 1, pp. 187–204, Feb. 2025.

[77] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.

[78] A. Al-Dujaili and S. Suresh, "A naive multi-scale search algorithm for global optimization problems," *Inf. Sci.*, vol. 372, pp. 294–312, 2016.

[79] H. Xu et al., "Omni-swarm: A decentralized omnidirectional visual–inertial–UWB state estimation system for aerial swarms," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3374–3394, Dec. 2022.

[80] C. Qi, T. Ma, Y. Li, Y. Ling, Y. Liao, and Y. Jiang, "A multi-AUV collaborative mapping system with bathymetric cooperative active SLAM algorithm," *IEEE Internet Things J.*, vol. 12, no. 9, pp. 12441–12452, May 2024.

[81] T. Qiu, Y. Li, and X. Feng, "Optimal broadcast scheduling algorithm for a multi-AUV acoustic communication network," *IEEE/ACM Trans. Netw.*, vol. 31, no. 5, pp. 2058–2069, Oct. 2023.

[82] J. Tan, I. Torroba, Y. Xie, and J. Folkesson, "Data-driven loop closure detection in bathymetric point clouds for underwater SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 3131–3137.

[83] D. Sowers, E. Lobecker, L. A. McKenna, E. Rose, J. Jacklyn, and M. Malik, "Mapping data acquisition and processing report, cruise EX-14-04 Leg 1 : Ship shakedown & patch test & exploration New England Seamounts (mapping), Aug. 9 - Aug. 29, 2014 N. Kingstown, RI - N. Kingstown, RI," USA, National Oceanic and Atmospheric Administration., Office of Ocean Exploration and Research, 2015. [Online]. Available: http://doi.org/10.7289/V5QN64RZ

[84] L. A. McKenna and B. R. Kennedy, "Mapping data acquisition and processing report, cruise EX-14-04 Leg III : Exploring Atlantic canyons and seamounts (ROV and mapping), Sep. 16 to Oct. 7, 2014 Baltimore, MD - N. Kingston, RI," USA, National Oceanic and Atmospheric Administration., Office of Ocean Exploration and Research, 2015. [Online]. Available: http://doi.org/10.7289/V5/MDR-OEREX1404L3

[85] I. Torroba, M. Cella, A. Terán, N. Rolleberg, and J. Folkesson, "Online stochastic variational Gaussian process mapping for large-scale bathymetric SLAM in real time," *IEEE Robot. Autom. Lett.*, vol. 8, no. 6, pp. 3150–3157, Jun. 2023.

[86] J. Wang, T. Liu, Q. Liu, X. He, H. Luo, and W. He, "Compression ratio modeling and estimation across error bounds for lossy compression," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 7, pp. 1621–1635, Jul. 2020.

[87] C. Roman and H. Singh, "Consistency based error evaluation for deep sea bathymetric mapping with robotic vehicles," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 3568–3574.

[88] Y. Tian and J. P. How, "Spectral sparsification for communication-efficient collaborative rotation and translation estimation," *IEEE Trans. Robot.*, vol. 40, pp. 257–276, 2024.

Qianyi Zhang (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in naval architecture and ocean engineering from Harbin Engineering University, Harbin, China, in 2019 and 2022, respectively. He is currently working toward the Ph.D. degree in robotics with the Robotics Program, Korea Advanced Institute of Science and Technology, Daejeon, South Korea.

His research interests include underwater perception, terrain-aided navigation, and bathymetric SLAM.

Jinwhan Kim (Member, IEEE) received the B.S. and M.S. degrees in naval architecture and ocean engineering from Seoul National University, Seoul, South Korea, in 1993 and 1995, respectively, and the Ph.D. degree in aeronautics and astronautics from Stanford University, Stanford, CA, USA, in 2007.

He previously worked with the Korea Research Institute of Ships and Ocean Engineering, Daejeon, South Korea, as a Researcher. He is currently a Professor with the Department of Mechanical Engineering, KAIST, Daejeon. His research interests include mobile robotics, unmanned systems, and vehicle intelligence.