IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

An Integrated Hierarchical Approach for Real-time Mapping with Gaussian Mixture Model

Yuan Gao and Wei Dong

***Abstract*—To achieve effective collaboration of multiple robots, it requires efficient exchanges of map information. As directly exchanging generally used depth map requires high communication bandwidth, it is practical to enhance the efficiency using map compression techniques based on Gaussian mixture models. Currently, parameters of the Gaussian mixture model are mostly computed using the expectation-maximization algorithm. It is time consuming as it has to iteratively update parameters by traversing all points in a point cloud converted from the depth map, and it is not suitable for real-time applications. Other methods directly segment the point cloud into grids and then perform a single Gaussian parameter estimation for each grid. They achieve real-time compression but generate parameter sensitive results. To tackle issues above, we improve compression methods with an integrated hierarchical approach. First, the points are clustered hierarchically and efficiently by K-means, generating coarse clusters. Then, each cluster is further hierarchically clustered by expectation-maximization algorithm for accuracy enhancement. After each clustering process, an evaluation index for ensuring accuracy and preventing overfitting is calculated to determine whether pruning or retention of newly generated clusters is appropriate. At last, parameters of each Gaussian distribution in the model are estimated by points in a corresponding cluster. Experiments conducted in various environments demonstrate that our approach improves computing efficiency by over 79 times compared to the state-of-the-art approach.**

***Index Terms*—Depth map compression, Gaussian mixture model, expectation-maximization algorithm, K-means.**

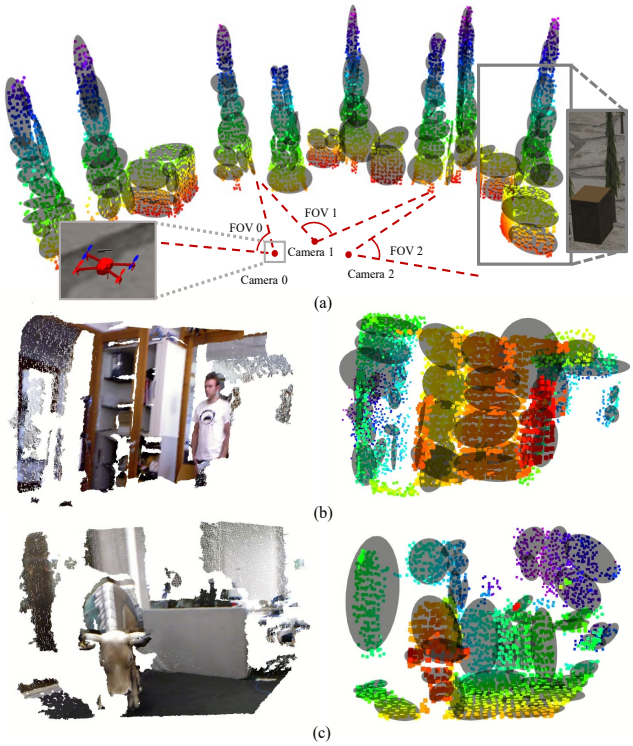

Fig. 1. Demonstration of GMM-based mapping. (a) A Gazebo simulation environment is collaboratively observed by three robots. The black ellipsoids represent Gaussian distributions. (b, c) GMM mapping for the real world.

I. Introduction

EXCHANGING map information is necessary for effective collaboration between multiple robots. By obtaining map information from different positions and angles, a multi-robot system can effectively enhance its perception ability compared to a single robot [1]. The sharing of map information between robots improves their robustness of movement in complex and unknown environments [2]. Depth map, a typical representation of map information, contains millions of pixels in each frame, creating challenges for robots to share them in real-time with limited communication bandwidth [3].

To improve the efficiency of exchanges, compressing depth maps is the most convenient method due to limitations of hardware devices. An effective technique for this task is using Gaussian mixture model (GMM), which represents continuous three-dimensional space and has desirable properties such as low memory requirements and anti-interference [4]. Consequently, it's appropriate to apply GMM, combined with several Gaussian distributions, to compress depth maps for real-time data transmission [3, 5]. By reconstructing depth maps based on received GMM parameters, global mapping is available [2].

Nevertheless, using the most traditional expectation-maximization algorithm to compute parameters of GMM (EM-GMM [6]) is extremely time-consuming because it estimates latent variables and updates Gaussian parameters by traversing all samples in each iteration. EM-GMM can be regarded as a clustering algorithm, as it clusters points in point clouds transformed from depth maps with various Gaussian distributions. Considering the replacement of EM-GMM, K-means [7] generates clustering results that are similar to EM-GMM's and is typically used for a fast initialization of EM-GMM [10, 11]. Despite this, since compression based on GMM is very sensitive to clustering accuracy, K-means is not applicable to GMM mapping due to its low accuracy. Other efficient

Manuscript received: May, 7, 2023; Revised: July, 23, 2023; Accepted August, 21, 2023. This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments. This work was supported in part by the National Natural Science Foundation of China Grant 51975348 and in part by Shanghai Rising-Star Program under Grant 22QA1404400. *(Corresponding author: Wei Dong.)*

All authors are with the State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: GaoY-23@sjtu.edu.cn; dr.dongwei@sjtu.edu.cn).

Digital Object Identifier (DOI): see top of this page.

clustering algorithms (e.g., density-based spatial clustering of applications with noise (DBSCAN) [8], ordering points to identify the clustering structure (OPTICS) [9], etc.) are also inappropriate for accurate compression. Therefore, the main problem of accurate GMM mapping is high time consumption.

In order to reduce time consumption, Eckart et al. [4] propose a top-down structure using EM-GMM. The algorithm's time complexity is reduced significantly but may lead to overfitting. Chen et al. [18] apply graph methods to improve the efficiency of GMM construction and point cloud registration technology. Gao et al. [19] reduce the number of degrees of freedom (DOFs) for all unknown GMM parameters to save memory use and reduce computational cost. In [20], hierarchical Gaussian mixture model (HGMM) is proposed for speaker verification. Shobhit et al. [21] propose a hierarchical structure and information theory techniques to construct a delicate global map. However, because of using optimization procedures to directly process large numbers of points, methods above still remain the problem of highly time consuming.

To improve computational efficiency of optimization procedures, some researchers reduce the number of points to be processed. Aditya et al. [22] utilize traditional EM-GMM algorithm to build GMM maps, i.e., compression with GMM. Their method ignores the space that have already been observed, so the number of points in each frame is small. Considering possible errors of GMM mapping, the method is not appropriate for high-precision mapping applications without eliminating the errors with multiple observations from similar perspectives.

Recognizing the drawbacks of reducing processed points, some researchers propose methods that avoid using optimization procedures. Saarinen at el. [10] introduce a novel 3D spatial representation that combines advantages of normal distributions transform (NDT) maps and occupancy grid maps. It partitions 3D space into voxels and generates one Gaussian distribution of points in each voxel. In [23], GMM is also constructed using pixels and then iteratively refined if neighbouring pixels represent a same surface in the environment. For computational and memory efficiency, [24] uses a single-pass to refine Gaussian distributions generated from scanlines, which are rows of a depth map. Unfortunately, the above methods require setting several geometric parameters based on the complexity of depth maps, which would make the algorithm inconvenient to use. Furthermore, different from using optimization procedures, setting improper geometric parameters in a particular environment can easily result to the decrease in the accuracy of GMM mapping. As a result, to avoid parameter sensitivity of GMM mapping, optimization procedures are irreplaceable.

To tackle the problem of highly time-consuming of using optimization procedures to process a large number of points, we improve the top-down structure with an integrated hierarchical approach. For real-time processing, K-means is first employed for efficient hierarchical clustering and the points are rapidly clustered into multiple smaller clusters. As the number of points in each cluster decreases, EM-GMM is used to achieve accurate clustering and compact Gaussian distribution generation. For further ensuring accuracy and preventing over-

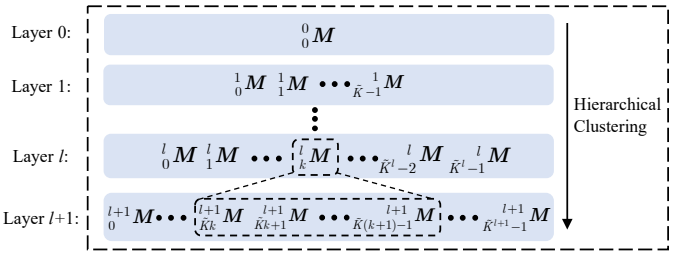

Fig. 2. Hierarchical Clustering Structure.

fitting, we define an evaluation index to terminate clustering process. If newly generated Gaussian distributions lead to low accuracy improvement and high model complexity, the distributions are judged to require deletion.

II. Integrated Hierarchical Clustering

In this section, an integrated hierarchical clustering approach (IH-GMM) is described. In Section II-A, we present a hierarchical structure. Then, we describe the specific procedures of EM-GMM and K-means in Section II-B and their integration in Section II-C. The time complexity of integrated hierarchical clustering is discussed in Section II-D.

A. Hierarchical Clustering Structure

A hierarchical clustering structure can reduce time complexity of computation. If directly partitioning N points into K clusters with t optimization iterations, the time complexity is $O(tdNK)$, where d is the number of operations required to compute the distance between cluster centers and points. Supposing a hierarchical structure with L layers clusters points into $\tilde{K}$ clusters of equal size at one time, the number of clusters finally obtained would be $K = \tilde{K}^L$, and the time complexity of the whole procedure is $O(\sum_{l=1}^{L} \tilde{K}^l td\frac{N}{\tilde{K}^l}\tilde{K})$, i.e., $O(tdN\tilde{K}\log_{\tilde{K}} K)$. It is easy to prove that when $K \gg \tilde{K} \geq 2$, $tdNK \gg tdN\tilde{K}\log_{\tilde{K}} K$.

As illustrated in Fig 2, a point matrix is defined as ${}^{l}_{k}\boldsymbol{M} = \left[{}^{l}_{k}\boldsymbol{m}_0, \ldots, {}^{l}_{k}\boldsymbol{m}_{\tilde{n}}, \ldots, {}^{l}_{k}\boldsymbol{m}_{\tilde{N}-1}\right]^T$ in hierarchical structure, where l denotes the layer of the matrix, k denotes the ordinal number of the matrix and ${}^{l}_{k}\boldsymbol{m}_{\tilde{n}}$ is the $\tilde{n}$th 3D coordinate vector of a point.

Clustering points in matrix ${}^{l}_{k}\boldsymbol{M}$, $\tilde{K}$ clusters would be obtained and their correspondingly point matrices are denoted as ${}^{l+1}_{\tilde{K}k}\boldsymbol{M}, {}^{l+1}_{\tilde{K}k+1}\boldsymbol{M}, \ldots, {}^{l+1}_{\tilde{K}(k+1)-1}\boldsymbol{M}$. The clustering is calculated by an clustering algorithm as follows:

$$ {}^{l}_{k}\boldsymbol{i}_{\tilde{N}\times 1} = c\left({}^{l}_{k}\boldsymbol{M}\right) \tag{1} $$

where $c(\cdot)$ represents the algorithm of clustering points and ${}^{l}_{k}\boldsymbol{i}_{\tilde{N}\times 1}$ is a clustering vector. Its entry ${}^{l}_{k}i_{\tilde{n}}$ indicates the ordinal number of the cluster which ${}^{l}_{k}\boldsymbol{m}_{\tilde{n}}$ belongs to, so ${}^{l}_{k}i_{\tilde{n}} \in \{0, 1, \ldots, \tilde{K}-1\}$.

New generated $\tilde{k}$th cluster by algorithm $c(\cdot)$ is donated as a set ${}^{l}_{k}C_{\tilde{k}}$. If the corresponding entry ${}^{l}_{k}i_{\tilde{n}}$ of $\tilde{n}$th coordinate vector satisfies ${}^{l}_{k}i_{\tilde{n}} = \tilde{k}$, ${}^{l}_{k}\boldsymbol{m}_{\tilde{n}}$ belongs to the $\tilde{k}$th set, i.e.

$$ {}^{l}_{k}\boldsymbol{m}_{\tilde{n}} \in \begin{cases} {}^{l}_{k}C_0 & , {}^{l}_{k}i_{\tilde{n}} = 0 \\ \vdots & \\ {}^{l}_{k}C_{\tilde{K}-1} & , {}^{l}_{k}i_{\tilde{n}} = \tilde{K}-1 \end{cases} \tag{2} $$

According to ${}_k^l C_{\tilde{k}}$, $\tilde{k}$th point matrix ${}_{\tilde{K}k+\tilde{k}}^{l+1}\boldsymbol{M}$ generated by clustering ${}_k^l\boldsymbol{M}$ can be defined as

$$ {}_{\tilde{K}k+\tilde{k}}^{l+1}\boldsymbol{M} = \left[{}_{\tilde{K}k+\tilde{k}}^{l+1}\boldsymbol{m}_0, \ldots, {}_{\tilde{K}k+\tilde{k}}^{l+1}\boldsymbol{m}_{\tilde{n}}, \ldots \right]^T \tag{3} $$

where ${}_{\tilde{K}k+\tilde{k}}^{l+1}\boldsymbol{m}_{\tilde{n}} \in {}_k^l C_{\tilde{k}}$.

B. Clustering by EM-GMM and K-means

EM-GMM and K-means are applied for clustering points. We define $\boldsymbol{M} = \left[\boldsymbol{m}_0, \ldots, \boldsymbol{m}_{\tilde{n}}, \ldots, \boldsymbol{m}_{\tilde{N}-1}\right]^T$ as the input matrix and $\boldsymbol{M}$ is clustered into $\tilde{K}$ clusters.

1) Clustering and GMM Construction with EM-GMM: It's supposed that points follow a GMM with $\tilde{K}$ distributions, whose probability density function [6] is formed as

$$ p(\boldsymbol{m}_{\tilde{\boldsymbol{n}}}) = \sum_{\tilde{k}=0}^{\tilde{K}-1} \pi_{\tilde{k}} \mathcal{N}\left(\boldsymbol{m}_{\tilde{\boldsymbol{n}}} \mid \boldsymbol{\mu}_{\tilde{k}}, \boldsymbol{\Sigma}_{\tilde{k}}\right) \tag{4} $$

where $\mathcal{N}\left(\boldsymbol{m}_{\tilde{\boldsymbol{n}}} \mid \boldsymbol{\mu}_{\tilde{k}}, \boldsymbol{\Sigma}_{\tilde{k}}\right)$ is a standard Gaussian distribution. $\boldsymbol{\mu}_{\tilde{k}}$ and $\boldsymbol{\Sigma}_{\tilde{k}}$ are mean vector and covariance matrix of $\tilde{k}$th distribution respectively. $\pi_{\tilde{k}}$ is the weight of $\tilde{k}$th distribution in GMM model with $\sum_{\tilde{k}=0}^{\tilde{K}-1} \pi_{\tilde{k}} = 1$.

EM-GMM algorithm [6] uses maximum likelihood function to construct GMM. In E(Estimation)-step, the probability that point $\boldsymbol{m}_n$ belongs to $\tilde{k}$th distribution, also called latent variable, is defined as

$$ \gamma_{\tilde{n},\tilde{k}} = \frac{\pi_{\tilde{k}} \mathcal{N}\left(\boldsymbol{m}_{\tilde{n}} \mid \boldsymbol{\mu}_{\tilde{k}}, \boldsymbol{\Sigma}_{\tilde{k}}\right)}{\sum_{\tilde{k}=0}^{\tilde{K}-1} \pi_{\tilde{k}} \mathcal{N}\left(\boldsymbol{m}_{\tilde{n}} \mid \boldsymbol{\mu}_{\tilde{k}}, \boldsymbol{\Sigma}_{\tilde{k}}\right)} \tag{5} $$

In M(Maximization)-step, the parameters are updated by

$$ \begin{aligned} \boldsymbol{\mu}_{\tilde{k}}^{\text{new}} &= \frac{1}{\tilde{N}_{\tilde{k}}} \sum_{\tilde{n}=0}^{\tilde{N}-1} \gamma_{\tilde{n},\tilde{k}} \boldsymbol{m}_{\tilde{n}} \qquad \pi_{\tilde{k}}^{\text{new}} = \frac{\tilde{N}_{\tilde{k}}}{\tilde{N}} \\ \boldsymbol{\Sigma}_{\tilde{k}}^{\text{new}} &= \frac{1}{\tilde{N}_{\tilde{k}}} \sum_{\tilde{n}=0}^{\tilde{N}-1} \gamma_{\tilde{n},\tilde{k}} \left(\boldsymbol{m}_{\tilde{n}} - \boldsymbol{\mu}_{\tilde{k}}^{\text{new}}\right) \left(\boldsymbol{m}_{\tilde{n}} - \boldsymbol{\mu}_{\tilde{k}}^{\text{new}}\right)^T \end{aligned} \tag{6} $$

where $\tilde{N}_{\tilde{k}} = \sum_{\tilde{n}=0}^{\tilde{N}-1} \gamma_{\tilde{n},\tilde{k}}$. By repeating (5) and (6), the optimal Gaussian parameters can be obtained when the algorithm converges. The point data $m_{\tilde{n}}$ is clustered into a set in which its probability $\gamma_{\tilde{n},\tilde{k}}$ is maximum, i.e.

$$ i_{\tilde{n}} = \arg\max_{\tilde{k}} \gamma_{\tilde{n},\tilde{k}} \tag{7} $$

2) Fast Clustering with K-means: K-means [7] is used for fast clustering of points. K-means first initializes $\tilde{K}$ centers. In E-step, it calculates the squares of distances from each point to all centers, i.e.

$$ D_{\tilde{n},\tilde{k}} = (\boldsymbol{m}_{\tilde{n}} - \boldsymbol{a}_{\tilde{k}})^T (\boldsymbol{m}_{\tilde{n}} - \boldsymbol{a}_{\tilde{k}}) \tag{8} $$

where $D_{\tilde{n},\tilde{k}}$ is the square of distance between point $\boldsymbol{m}_{\tilde{n}}$ and center $\boldsymbol{a}_{\tilde{k}}$. The latent variable of K-means is defined as $\boldsymbol{i}_{\tilde{N}\times 1} = \left[i_0, \ldots, i_{\tilde{n}}, \ldots, i_{\tilde{N}-1}\right]^T$, where $i_{\tilde{n}}$ represents the ordinal number of a center, whose distance to point $\boldsymbol{m}_{\tilde{n}}$ is shortest (i.e., $i_{\tilde{n}} = \arg\min_{\tilde{k}} D_{\tilde{n},\tilde{k}}$). In M-step, $\boldsymbol{a}_{\tilde{k}}$ can be updated as

$$ \boldsymbol{a}_{\tilde{k}}^{\text{new}} = \frac{1}{u(\boldsymbol{i},\tilde{k})} \sum_{i_{\tilde{n}}=\tilde{k}} \boldsymbol{m}_{\tilde{n}} \tag{9} $$

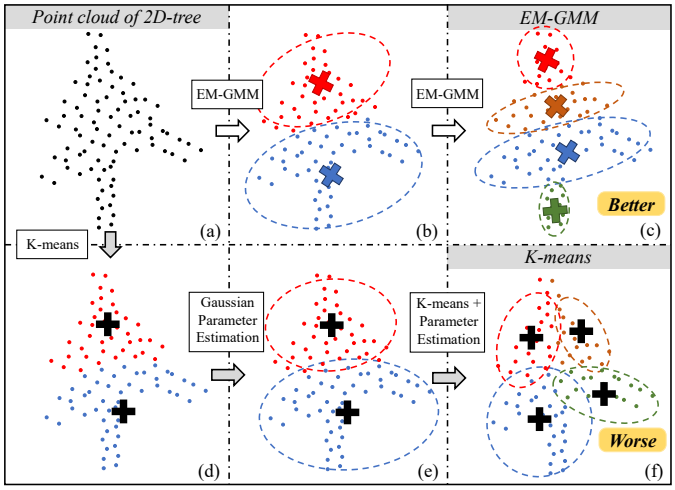

Fig. 3. Effect of EM-GMM and K-means on GMM construction.

where $u(\boldsymbol{i},\tilde{k})$ represents the number of entry $\tilde{k}$ in vector $\boldsymbol{i}$. $\boldsymbol{a}_{\tilde{k}}$ would converge to optimal value by repeating (8) and (9).

Compared to EM-GMM, parameter estimation should be carried out after K-means converges. Supposing that points in each cluster follow a Gaussian distribution, $\boldsymbol{a}_{\tilde{k}}$ is the distribution's mean $\boldsymbol{\mu}_{\tilde{k}}$, the intensity $\pi_{\tilde{k}}$ of such distribution is the ratio between the number of points contained in $\tilde{k}$th cluster and the total number of points (i.e., $u(\boldsymbol{i},\tilde{k})/\tilde{N}$) and covariance matrix $\boldsymbol{\Sigma}_{\tilde{k}}$ is the second order moment of the points in $\tilde{k}$th cluster.

C. Integration of EM-GMM and K-means

Computational consumption of a iterative optimization algorithm is related to its time complexity. Defining the time complexity of K-means and EM-GMM as $O(t_{\mathcal{K}} d_{\mathcal{K}} \tilde{N}\tilde{k})$ and $O(t_{\mathcal{E}} d_{\mathcal{E}} \tilde{N}\tilde{k})$. They both use the idea of EM algorithm, but the latent variable of K-means is simpler than that of EM-GMM, so the numbers of optimization iterations of K-means are normally lower than EM-GMM (i.e., $t_{\mathcal{K}} < t_{\mathcal{E}}$). Referring to (5), (6), (8) and (9), EM-GMM is much more complicated than K-means to execute both E-steps and M-steps (i.e., $d_{\mathcal{K}} < d_{\mathcal{E}}$). Therefore, $t_{\mathcal{K}} d_{\mathcal{K}} \tilde{N}\tilde{k} < t_{\mathcal{E}} d_{\mathcal{E}} \tilde{N}\tilde{k}$.

As shown in Fig. 3, the accuracy of hierarchical clustering when using EM-GMM and K-means is compared. Comparing Fig. 3 (b) and Fig. 3 (e), EM-GMM and K-means generate almost same results when processing large clusters. After a further clustering, EM-GMM performs better in Fig. 3 (c) than K-means in Fig. 3 (f). As a result, it is reasonable to replace EM-GMM with K-means to segment the large clusters due to their similar performances and when processing smaller clusters, using EM-GMM to ensure accuracy is necessary.

To define the concept of large clusters, a threshold value N_t is predesigned. When the number of points $\tilde{N}$ is higher than N_t, K-means is used to perform clustering. Conversely, EM-GMM is used. Refining the clustering process in (1), one can obtain

$$ {}_k^l\boldsymbol{i}_{\tilde{N}\times 1} = c\left({}_k^l\boldsymbol{M}\right) = \begin{cases} c_{\mathcal{K}}\left({}_k^l\boldsymbol{M}\right), \tilde{N} > N_t \\ c_{\mathcal{E}}\left({}_k^l\boldsymbol{M}\right), \tilde{N} \le N_t \end{cases} \tag{10} $$

where $c_{\mathcal{K}}(\cdot)$ is a clustering process by K-means, and $c_{\mathcal{E}}(\cdot)$ uses EM-GMM.

D. Time Complexity of Integrated Hierarchical Clustering

Supposing the structure segments points into clusters of equal size at each time, the time complexity of a hierarchical clustering structure using only EM-GMM algorithm is $O(\sum_{l=1}^{L}\tilde{K}^l t_{\mathcal{E}} d_{\mathcal{E}} \frac{N}{\tilde{K}^l}\tilde{K})$, i.e., $O(N\tilde{K}Lt_{\mathcal{E}}d_{\mathcal{E}})$. By integrating K-means into such structure, time complexity is reduced sharply. It is defined that when clustering points into l_tth layer, sizes of clusters are below the threshold value N_t (i.e., $N/n^{l_t} < N_t < N/n^{l_t-1}$). The entire procedure's time complexity is $O(\sum_{l=1}^{l_t-1}\tilde{K}^l t_{\mathcal{K}} d_{\mathcal{K}} \frac{N}{\tilde{K}^l}\tilde{K} + \sum_{l=l_t}^{L}\tilde{K}^l t_{\mathcal{E}} d_{\mathcal{E}} \frac{N}{\tilde{K}^l}\tilde{K})$, i.e., $O(N\tilde{K}Lt_{\mathcal{E}}d_{\mathcal{E}} - N\tilde{K}(l_t-1)(t_{\mathcal{E}}d_{\mathcal{E}} - t_{\mathcal{K}}d_{\mathcal{K}}))$. Considering that $t_{\mathcal{E}}d_{\mathcal{E}} - t_{\mathcal{K}}d_{\mathcal{K}} > 0$, the computing efficiency is reduced by the integration of K-means and EM-GMM.

III. Evaluation and Termination of Clustering

One drawback of the integrated hierarchical structure described in Section II is that the clustering process cannot be terminated reasonably by only predesigning number of layers L. During continuous clustering, when the accuracy of GMM mapping reaches a satisfactory level, the process should be promptly terminated in order to prevent over-fitting and reduce computational consumption. Therefore, based on integrated hierarchical structure, an evaluation index can be applied to determine whether to terminate the clustering, i.e., to judge the reasonableness of newly generated Gaussian distributions.

Currently, several methods have been proposed to evaluate the reasonableness of GMM (e.g., Akaike information criterion (AIC) [12, 13], Bayesian information criterion (BIC) [14], approximate weight of evidence criterion (AWE) [15], normalized entropy criterion (NEC) [16], integrated classification criterion (ICL) [17], etc.). BIC and AIC are both typical information criteria, which are used to quantify the balance between the complexity of model and the ability of model to represent points (i.e., maximum likelihood function). AWE, NEC and ICL further quantify the degree of separation of all Gaussian distributions.

However, the criteria above focus on constructing a compact probabilistic model, ignoring the requirement of reconstruction accuracy when mapping with GMM. As a result, a specialized evaluation for GMM mapping is necessary. The novel evaluation index is defined in Section III-A, and the termination strategy of clustering by the index is shown in Section III-B.

A. Overfitting and Accuracy Evaluation

The evaluation considers maximum likelihood function, model complexity and reconstruction accuracy of GMM.

1) Maximum Likelihood Function: For GMM, its maximum likelihood function [25], the maximum objective of EM-GMM algorithm, is related to all points and $\tilde{K}$ Gaussian distributions as follows

$$H = \sum_{\tilde{n}=1}^{\tilde{N}} \log\left(\sum_{\tilde{k}=1}^{\tilde{K}} \pi_{\tilde{k}} \mathcal{N}\left(\boldsymbol{m}_{\tilde{n}} \mid \boldsymbol{\mu}_{\tilde{k}}, \boldsymbol{\Sigma}_{\tilde{k}}\right)\right) \tag{11}$$

For convenience of description, $H(\boldsymbol{M}, \boldsymbol{i})$ is defined as a maximum likelihood function when points $\boldsymbol{M}$ are clustered by vector $\boldsymbol{i}$. The corresponding Gaussian parameters, including $\pi_{\tilde{k}}$, $\boldsymbol{\mu}_{\tilde{k}}$, and $\boldsymbol{\Sigma}_{\tilde{k}}$, are estimated in each cluster. $H(\boldsymbol{M}, \boldsymbol{i})$ quantifies the compactness of constructed GMM.

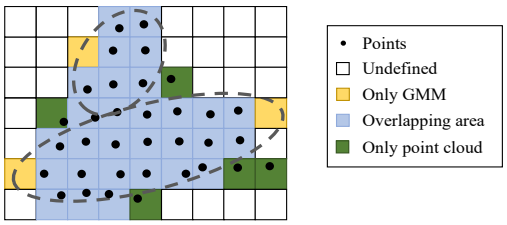

Fig. 4. Division of 3D space with grids. It is worth noting that only the grid whose center is contained by a Gaussian ellipsoid and contains no point is considered being only occupied by GMM. A grid contains a point which is covered by Gaussian ellipsoid is considered as an overlapping area. If not, it is identified as only occupied by points.

2) Model Complexity: Model complexity is quantified to prevent overfitting, which means using an overly complex model to represent points. Generally, more points require more Gaussian distributions to compress depth maps. Therefore, model complexity is proportional to the number of Gaussian distributions and inversely proportional to the number of points. We define the complexity of GMM as ratio of the number of Gaussian distributions to the number of points, i.e., $\tilde{K}/\tilde{N}$. In clustering vector $\boldsymbol{i}$, $\tilde{K}-1$ is its largest entry and $\tilde{N}$ is the length of $\boldsymbol{i}$. Therefore, the model complexity can be written as

$$C(\boldsymbol{i}) = \frac{m(\boldsymbol{i})+1}{v(\boldsymbol{i})} \tag{12}$$

where $m(\boldsymbol{i})$ is the largest entry in $\boldsymbol{i}$ and $v(\boldsymbol{i})$ is its length.

3) Reconstruction Accuracy: To quantify the accuracy of Gaussian distributions calculated by K-means and EM-GMM, we use grids to partition 3D space as shown in Fig. 4. Coverage rate of GMM is defined as

$$R_c(\boldsymbol{M}, \boldsymbol{i}) = \frac{A(\boldsymbol{M}, \boldsymbol{i})}{A(\boldsymbol{M}, \boldsymbol{i}) + A_{\mathcal{P}}(\boldsymbol{M}, \boldsymbol{i})} \tag{13}$$

where $A(\boldsymbol{M}, \boldsymbol{i})$ is the number of grids which occupied by overlapping area (i.e., blue grids in Fig. 4) and $A_{\mathcal{P}}(\boldsymbol{M}, \boldsymbol{i})$ is volume of space only containing points (i.e., green grids). Effect of coverage rate is similar to that of maximum likelihood function, but it's more intuitive. When points are not covered by Gaussian distributions, both maximum likelihood function and coverage rate are reduced. Since maximum likelihood function is required to be calculated by EM-GMM algorithm, we do not calculate coverage rate additionally only for calculating the evaluation index.

To describe the similarity between the space occupied by the GMM and the space occupied by points, reconstruction accuracy is defined as

$$R(\boldsymbol{M}, \boldsymbol{i}) = \frac{2A(\boldsymbol{M}, \boldsymbol{i})}{2A(\boldsymbol{M}, \boldsymbol{i}) + A_{\mathcal{G}}(\boldsymbol{M}, \boldsymbol{i}) + A_{\mathcal{P}}(\boldsymbol{M}, \boldsymbol{i})} \tag{14}$$

where $A_{\mathcal{G}}(\boldsymbol{M}, \boldsymbol{i})$ is the volume of space which contains only GMM ellipsoids (i.e., yellow grids in Fig. 4). We calculate $A_{\mathcal{G}}(\boldsymbol{M}, \boldsymbol{i})$ with 3Σ-probabilistic bound.

4) Evaluation Index: When clustering points $\boldsymbol{M}$ with vector $\boldsymbol{i}$, the evaluation index $E(\boldsymbol{M}, \boldsymbol{i})$ is defined as follows

$$E(\boldsymbol{M}, \boldsymbol{i}) = \lambda_H H(\boldsymbol{M}, \boldsymbol{i}) - \lambda_C C(\boldsymbol{i}) + \lambda_R R(\boldsymbol{M}, \boldsymbol{i}) \tag{15}$$

where λ_H, λ_C, and λ_R determine the weight of each term.

B. Termination of Clustering Process

After clustering ${}^{l}_{k}\boldsymbol{M}$, the evaluation index before and after is compared to determine whether the clustering is reasonable. If the clustering of ${}^{l}_{k}\boldsymbol{M}$ is unreasonable, all the points are put into one cluster and other clusters are empty. The new clustering vector is defined as

$$ {}^{l}_{k}\boldsymbol{i} = \begin{cases} {}^{l}_{k}\boldsymbol{i} & , E({}^{l}_{k}\boldsymbol{M}, {}^{l}_{k}\boldsymbol{i}) > E({}^{l}_{k}\boldsymbol{M}, \mathbf{0}_{\tilde{N}\times 1}) \\ \mathbf{0}_{\tilde{N}\times 1} & , E({}^{l}_{k}\boldsymbol{M}, {}^{l}_{k}\boldsymbol{i}) \leq E({}^{l}_{k}\boldsymbol{M}, \mathbf{0}_{\tilde{N}\times 1}) \end{cases} \tag{16} $$

Continuously clustering points and segmenting clusters until no new cluster can be generated, the GMM mapping is completed. For a hierarchical structure clustering points into Lth layer, the complexity of the algorithm is not affected by adding such termination.

IV. GMM-based Compression Algorithm

In this part, we describe the whole operation process of the algorithm, including the points pretreatment in Section IV-A and algorithm implementation in Section IV-B.

A. Pretreatment

For down-sampling, points need to be voxel filtered before clustering. Voxel filter constructs grid maps first, and then calculates the mean of all points in each grid as a new point. In this paper, the filtering uses the same grid size as 3D space partition use for calculating $R(\boldsymbol{M}, \boldsymbol{i})$. Since each grid contains a point in the filtered point cloud, the partition generated by filtered point cloud is the same as what generated by original point cloud. As a result, such down-sampling does not affect the estimation of reconstruction accuracy $R(\boldsymbol{M}, \boldsymbol{i})$.

The filtered points are likely to be discontinuous, as they are composed of multiple point clusters from different objects, such as trees and boxes (Fig. 1(a)). Clustering of such point clouds is highly time-consuming, as it requires large numbers of iterations if initialization is improper. Therefore, we segment original points with Euclidean segmentation. Such segmentation puts any two points whose distance is less than a threshold into one set. Our proposed integrated hierarchical approach is used to cluster points in each set which is computed by Euclidean segmentation.

B. GMM Mapping Algorithm

The GMM mapping algorithm is shown in Alg. 1. $\tilde{K}$ is set to be 2. By processing points in an original point cloud with voxel filtering and Euclidean segmentation, the input of algorithm $\boldsymbol{M}_0$ is obtained. In Line 2, we define a structure Z containing point matrix $\boldsymbol{M}$, Gaussian parameter Θ, evaluation index E and termination mark t. Specifically, E is calculated by points in $\boldsymbol{M}$ (Line 5) and t is the mark of whether $\boldsymbol{M}$ needs to be further clustered (Line 6). After selecting a structure $Z[k]$ needs to be clustered (Lines 9-11), its clustering vector $\boldsymbol{i}$ is calculated in Line 12. In Line 13, the value of the largest entry contained in the vector $\boldsymbol{i}$ is judged to determine whether the matrix $\boldsymbol{Z}[k].\boldsymbol{M}$ being clustered needs to be partitioned further, i.e. whether this clustering is reasonable. A segmentation of $Z[k].\boldsymbol{M}$ is done in Line 14 if $\boldsymbol{M}$ needs to be clustered. To reduce memory consumption, we fine-tuned the hierarchical structure into an incremental hierarchical structure by deleting the processed $Z[k]$ and pushing back new ones in $\boldsymbol{Z}$ (Lines 15-17). Since all clusters are still hierarchically clustered, the incremental structure will not affect the algorithm's complexity.

Algorithm 1 Integrated Hierarchical Clustering

Input: Pretreated Point Matrix $\boldsymbol{M}_0$
Output: GMM Parameters $\boldsymbol{\Theta} = [\Theta_0, \Theta_1, \dots]$

```
1:  procedure INTEGRATED_HIERARCHICAL_CLUSTERING(M_0)
2:      Z ≜ {Point Matrix M, Gaussian Parameter Θ,
             Evaluation Index E, Termination Mark t}
3:      Z.M ← M_0
4:      Z.Θ ← GaussianEstimate (M_0)
5:      Z.E ← Evaluation (M_0, 0)
6:      Z.t ← 0
7:      Z.pushback(Z)
8:      for k ← 0; k < Z.size() − 1; k ← k + 1
9:          if Z[k].t
10:             continue
11:         end if
12:         i ← Cluster (Z[k].M)
13:         if i.max() > 0
14:             {Z', Z''} ← Segment (Z[k].M, i)
15:             Z.delete(k)
16:             Z.pushback(Z')
17:             Z.pushback(Z'')
18:             k ← k − 1
19:         else
20:             Z[k].t ← 1
21:         end if
22:     end for
23:     for k ← 0; k < Z.size() − 1; k ← k + 1
24:         Θ[k] ← Z[k].Θ
25:     end for
26:     return Θ
27: end procedure
28: procedure CLUSTER (M)
29:     i ← K-means (M)
30:     if M.size() ≤ N_t
31:         i ← EM-GMM (M, i)
32:     end if
33:     if Termination_Judge (i, Z[k])
34:         i ← 0
35:     end if
36:     return i
37: end procedure
38: procedure TERMINATION_JUDGE (Z, i)
39:     E ← Evaluation (Z.M, i)
40:     if E ≤ Z.E
41:         return 1
42:     else
43:         return 0
44: end procedure
```

V. Experiments

In this section, we first test the compression rate of our approach in Section V-A. Then we compare the computational efficiency (Section V-B) and accuracy (Section V-C) of different methods. At last, the sensitivity and supplement of our approach are presented in Section V-D and Section V-E.

We choose traditional EM-GMM [6], K-means [7], H-EM-GMM [4], and H-K-means as reference methods, which are all iterative-optimization-based clustering algorithms. Among them, H-K-means is a hierarchical clustering algorithm that replaces EM-GMM with K-means to perform each cluster-

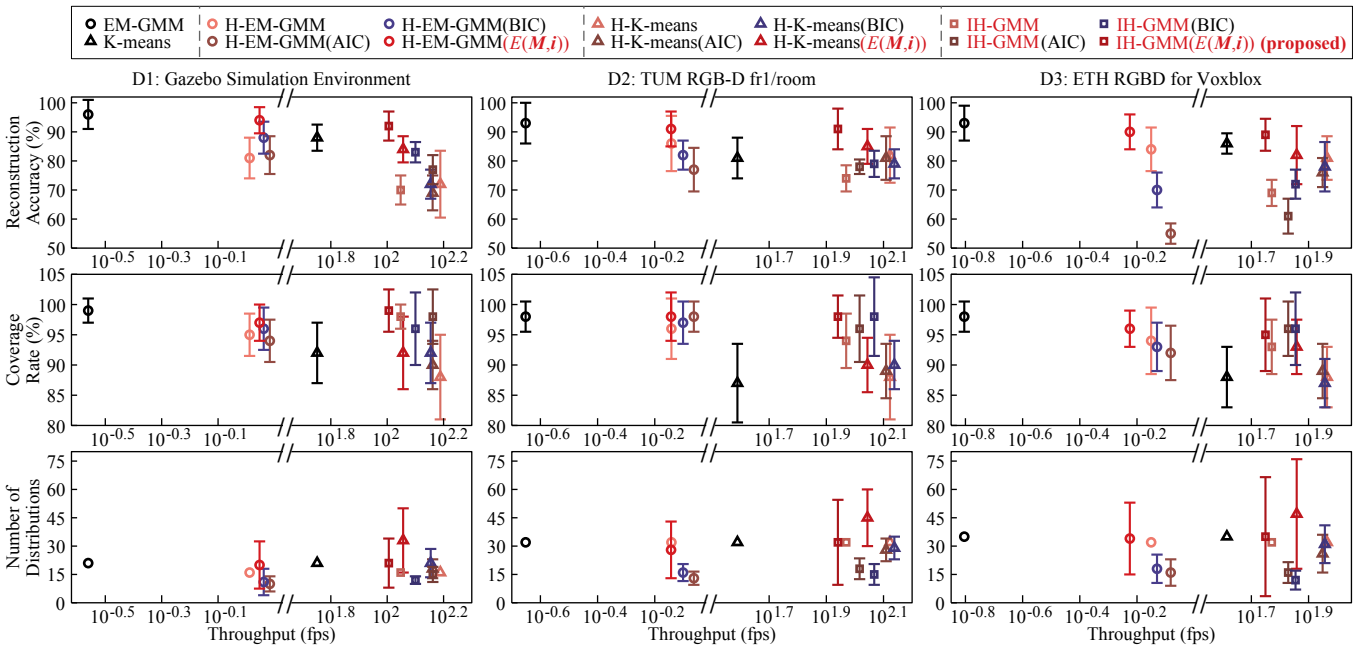

Fig. 5. The comparison between each method. EM-GMM [6], K-means [7], H-EM-GMM [4], H-K-means and our proposed structure (IH-GMM) with different termination evaluation index/criteria (e.i., AIC [12, 13], BIC [14] and $E(\boldsymbol{M}, \boldsymbol{i})$) are compared with each other.

TABLE I
COMPRESSION RATE OF PROPOSED METHOD

Data Set	Data Volume (KB) GMM	Data Volume (KB) Filtered Points	Compression Rate (%)
D1	2.2(+1.4, 1.4)	22.1(+5.5, 3.9)	**9.7**(+**13.6**, **5.6**)
D2	3.3(+2.2, 2.5)	13.8(+22.3, 10.6)	23.7(+9.4, 15.1)
D3	3.6(+3.8, 2.7)	28.9(+19.7, 12.9)	12.3(+9.1, 3.3)

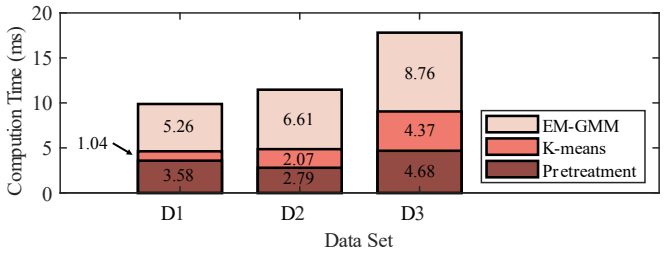

Fig. 6. Time allocation of proposed method.

ing process. To compare the performance of all evaluation index/criteria (AIC [12, 13], BIC [14] and $E(\boldsymbol{M}, \boldsymbol{i})$), H-EM-GMM and H-K-means are also improved with termination procedures. All the algorithms are tested using a simulation environment (D1: A Simulation environment in Gazebo) and real-world data sets (D2: TUM RGB-D fr1/room [26] and D3: ETH RGBD for Voxblox [27]) with same pretreatment. All experiments are performed on a laptop with Intel 12700H.

For each depth map, a 0.1m*0.1m*0.1m voxel filtering is conducted on points. The distance threshold of Euclidean segmentation is set to be 0.3m and the threshold of points' number to use EM-GMM (i.e., N_t) is 200. λ_H, λ_C, and λ_R are initialized to be 1, 150 and 100 respectively by trial and error approach. This parameter setting ensures that the algorithm can maximize computational efficiency and minimize the number of Gaussian distributions while maintaining accuracy. When using EM-GMM and K-means directly (i.e., without hierarchical structure), the number of Gaussian distributions is initialized to be equal to the average number of the proposed method. Distributions' number of H-EM-GMM and H-K-means is also predesigned similar to the average number.

The comparison between each method is shown in Fig. 5 in a form of Pareto Front [28]. The ranges overlie the bars represent the 3σ bounds of data.

A. Compression Rate

The compression rate of proposed method tested on different data sets is shown in Table I. The form of expression (e.g., **9.7**(+**13.6**, −**5.6**)) represents the mean (i.e., **9.7**), upper-limit (i.e., **9.7** + **13.6**) and lower-limit (i.e., **9.7** − **5.6**) of the data.

It can be noted that the compression rate of depth maps is not only related to the number of points, but rather it is also influenced by the complexity of the environment. For example, D2 contains the smallest number of points of 13.8KB, but it still requires multiple Gaussian distributions of 3.3KB for compression. Due to the complex environment in D2, the proposed algorithm spends a larger number of Gaussian distributions to achieve high compression accuracy, thus increasing the volume of GMM data.

B. Computational Efficiency

As shown in Fig. 5, most of the algorithms are more time consuming when compressing data in D2 and D3 due to higher complexity of depth maps from real-world environments. EM-GMM is the least computationally efficient, followed by H-EM-GMM. Neither of them can achieve real-time processing. H-K-means is the most efficient due to not using EM-GMM for clustering throughout the entire process. The proposed method achieves a speed of more than 50 fps (17.81 ms) when processing D3, which is almost 79 times better than H-EM-GMM algorithm. In D1 and D2, the improvement is 98 times and 120 times respectively, which is more remarkable.

Main steps of the compression involve pretreatment, hierarchical K-means and hierarchical EM-GMM. Referring to Table

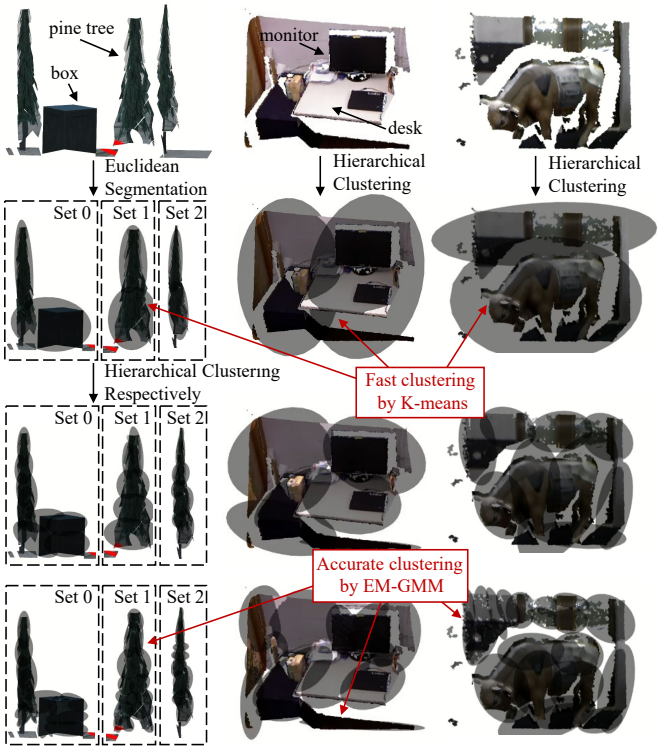

Fig. 7. Generating process of GMM mapping. From top to bottom, the point clouds are first coarsely clustered and then accurately clustered.

I and Fig. 6, the pretreatment time is basically proportional to the number of points. D3 takes the longest pretreatment time of 4.68ms, followed by D1 and D2. The reason for this is that voxel filtering is the most time-consuming part of pretreatment and each point is traversed a relatively fixed number of times. After that, it takes little time to process filtered points with Euclidean segmentation, thereby having an insignificant impact on the consuming time of pretreatment. EM-GMM takes the longest time in all data sets, accounting for 53%, 56%, and 49% of the total time respectively. However, it has a great advantage over the time spent by H-EM-GMM, which uses EM-GMM to perform coarse clustering processes.

It is worth noting that the clustering time of D2 is longer than that of D1 for both K-means and EM-GMM, but the number of points need to be clustered is lower than that of D1. The reason is that D2 is collected from the real-world and is more complex. More complex point clouds often require the algorithm to spend more iterations to attain convergence, making the compression for data in D2 more time-consuming. Another reason is that point clouds in D1 are always discontinuous, making it common practice to undertake Euclidean segmentation, decreasing the number of coarse clustering processes by K-means. Moreover, D2 requires more layers of accurate clustering to reach accuracy requirements.

C. Accuracy

The hierarchical generations of GMM maps are shown in Fig. 7. As can be seen, the proposed method compresses the depth map from rough to accurate.

To verify the performance of the evaluation index $E(\boldsymbol{M}, \boldsymbol{i})$ defined in our method, we compare it with AIC and BIC which are used to prevent over-fitting in Fig. 5. When using IH-GMM structure, proposed evaluation index $E(\boldsymbol{M}, \boldsymbol{i})$ can further enhance the GMMs' accuracy, performing better than AIC and BIC. Compared to extremely time-comsuming EM-GMM, the reconstruction accuracy of proposed algorithm is at most 4% lower, which is still at at a high level. Meanwhile, more Gaussian distributions are generated by using $E(\boldsymbol{M}, \boldsymbol{i})$ and it makes the the number of Gaussian distributions more adaptive. In D2, at least 6 Gaussian distributions are used for accurate compression of a depth map. In addition, due to the consideration of the maximum likelihood function in each index or criterion, the average coverage rates of GMM maps are all higher than 95%.

To test the impact of voxel filtering on accuracy of GMM maps, the proposed approach is used to compress point clouds without down-sampling. By setting N_t proportion to the number of points, the reconstruction accuracy of compressing point clouds without down-sampling tested on three data sets is 92 ± 5, 92 ± 7 and 89 ± 5 respectively. When processing down-sampled point clouds, the reconstruction accuracy is 92 ± 5, 91 ± 7 and 89 ± 6 (Fig. 5), which is close to the results above. This indicates that the voxel filtering has little effect on the accuracy with the parameter settings in this paper.

D. Sensitivity

The sensitivity of algorithm to λ_H, λ_C, and λ_R is analyzed in this subsection. Since the experimental results are greatly affected by data sets, the experimental approach is used for the analysis. As shown in Fig. 8, sensitivity analysis of N and $R(\boldsymbol{M}, \boldsymbol{i})$ are performed by adjusting each parameter individually around initial values. $S(\cdot)$ is the rate-of-change, e.g. $S(N) = (N - N_{\text{origin}})/N_{\text{origin}} \times 100\%$. When λ_H is larger than 0.5, the algorithm will not lead to over-fitting. Besides, an increase in λ_C leads to a slow decrease in N, while an increase in λ_R leads to an increase in $R(\boldsymbol{M}, \boldsymbol{i})$. The ranges of most variations are less than 2%, indicating the high stability of the proposed approach with respect to the parameters.

E. Supplement

Currently, our approach only compresses point clouds transformed from a single depth map acquired by the depth camera. For applications in large environments or fields of views, relevant experiments have not been performed.

When building GMM maps of large environments, the fusion, filtering and registration of Gaussian distributions should

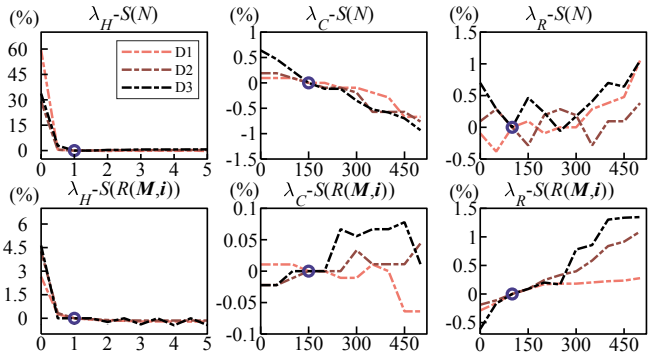

Fig. 8. Sensitivity analysis. It shows the rate-of-change of N and $R(\boldsymbol{M}, \boldsymbol{i})$ with respect to each parameter, where the abscissa corresponding to each purple circle represents the initial value of the parameter.

be performed, which haven't been tackled by current works. For cooperative observation with multi-robots, the registration of Gaussian distributions in different coordinate systems is also necessary to build a global map.

Currently, many deep learning based methods are used for GMM construction [29, 30]. Such methods tend to be more accurate as their objective function can be adapted. The large number of nodes also ensures that such methods have high degrees of freedom. In this paper, the proposed method improves computational efficiency and makes the number of Gaussian distributions adaptive.

VI. Conclusion

In this study, we propose an integrated hierarchical approach for accurate and real-time compression of map information, which utilizes a Gaussian mixture model to compress depth maps. With a hierarchical structure, computational consumption is reduced sharply. Such structure enables the integration of K-means and EM-GMM, using them for clustering points in different layers of the structure. By efficiently generating coarse GMMs with K-means and then refining them with EM-GMM, our approach realizes the integration of the two clustering algorithms. After each clustering process, a termination judgement based on an anti-overfitting and accuracy evaluation index is applied to prune redundant distributions for further accuracy insurance. Compared to existing algorithms, our approach improves efficiency by more than 79 times. Moreover, it can compress different depth maps with adaptive numbers of Gaussian distributions to maintain high fidelity. GMM maps have great potential in depth map compression and map parameterization. In the future, we plan to explore techniques for global GMM mapping with multi-robot collaboration, which can improve the robustness of path planning for multi-robot systems.

References

[1] S.-J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, "A Survey on Aerial Swarm Robotics," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 837–855, Aug. 2018.

[2] P. Peng, W. Dong, G. Chen, and X. Zhu, "Obstacle Avoidance of Resilient UAV Swarm Formation with Active Sensing System in the Dense Environment," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, Japan: IEEE, Oct. 2022, pp. 10529–10535.

[3] H. Dong et al., "MR-GMMapping: Communication Efficient Multi-Robot Mapping System via Gaussian Mixture Model," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3294–3301, Apr. 2022.

[4] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, "Accelerated Generative Models for 3D Point Cloud Data," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 5497–5505.

[5] M. Corah, C. O'Meadhra, K. Goel, and N. Michael, "Communication-Efficient Planning and Mapping for Multi-Robot Exploration in Large Environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1715–1721, Apr. 2019.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data Via the EM Algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, Sep. 1977.

[7] J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations," in *Proc. 5th Berkeley Symp. on Math. Statist. and Prob.*, vol. 10, no. 3, pp. 281–297, Mar. 2018.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231, Aug. 1996.

[9] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: ordering points to identify the clustering structure," *SIGMOD Rec.*, vol. 28, no. 2, pp. 49–60, Jun. 1999.

[10] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "3D normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, Dec. 2013.

[11] Y. Xu et al., "GAME: Gaussian Mixture Model Mapping and Navigation Engine on Embedded FPGA," in *2021 IEEE 29th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Orlando, FL, USA: IEEE, May 2021, pp. 60–68.

[12] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Automat. Contr.*, vol. 19, no. 6, pp. 716–723, Dec. 1974.

[13] H. Akaike, "Information Theory and an Extension of the Maximum Likelihood Principle," in *Breakthroughs in Statistics, S. Kotz and N. L. Johnson, Eds., in Springer Series in Statistics*. New York, NY: Springer New York, 1992, pp. 610–624.

[14] G. Schwarz, "Estimating the Dimension of a Model," *Ann. Statist.*, vol. 6, no. 2, Mar. 1978.

[15] J. D. Banfield and A. E. Raftery, "Model-Based Gaussian and Non-Gaussian Clustering," *Biometrics*, vol. 49, no. 3, p. 803, Sep. 1993.

[16] G. Celeux and G. Soromenho, "An entropy criterion for assessing the number of clusters in a mixture model," *Journal of Classification*, vol. 13, no. 2, pp. 195–212, Sep. 1996.

[17] C. Biernacki, G. Celeux, and G. Govaert, "Assessing a mixture model for clustering with the integrated completed likelihood," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 7, pp. 719–725, Jul. 2000.

[18] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovacevic, "Fast Resampling of Three-Dimensional Point Clouds via Graphs," *IEEE Trans. Signal Process.*, vol. 66, no. 3, pp. 666–681, Feb. 2018.

[19] G. Gao et al., "Reduced-Degrees-of-Freedom Gaussian-Mixture-Model Fitting for Large-Scale History-Matching Problems," *SPE Journal*, vol. 25, no. 01, pp. 037–055, Feb. 2020.

[20] M. Liu, E. Chang, and B. Dai, "Hierarchical Gaussian mixture model for speaker verification," in *7th International Conference on Spoken Language Processing (ICSLP 2002)*, ISCA, Sep. 2002, pp. 1353–1356.

[21] S. Srivastava and N. Michael, "Approximate continuous belief distributions for precise autonomous inspection," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Lausanne, Oct. 2016, pp. 74–80.

[22] A. Dhawale, X. Yang, and N. Michael, "Reactive Collision Avoidance Using Real-Time Local Gaussian Mixture Model Maps," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Oct. 2018, pp. 3545–3550.

[23] A. Dhawale and N. Michael, "Efficient Parametric Multi-Fidelity Surface Mapping," in *Robotics: Science and Systems XVI, Robotics: Science and Systems Foundation*, Jul. 2020.

[24] P. Z. X. Li, S. Karaman, and V. Sze, "Memory-Efficient Gaussian Fitting for Depth Images in Real Time," in *2022 International Conference on Robotics and Automation (ICRA)*, Philadelphia, PA, USA: IEEE, May 2022, pp. 8003–8009.

[25] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," *Int. Computer Science Institute*, vol. 4-510, p. 126, Apr. 1998.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 573–580.

[27] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC: IEEE, Sep. 2017, pp. 1366–1373.

[28] J. Delmerico and D. Scaramuzza, "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, Australia, 2018, pp. 2502-2509.

[29] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings,* 2014.

[30] K. Kinoshita, M. Delcroix, A. Ogawa, T. Higuchi and T. Nakatani, "Deep mixture density network for statistical model-based feature enhancement," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, 2017, pp. 251-255.