2022 IEEE International Conference on
Systems, Man, and Cybernetics (SMC)

Batch Learning Growing Neural Gas for Sequential Point Cloud Processing

Fernando Ardilla, Azhar Aulia Saputra, Naoyuki Kubota
Department of Mechanical Systems Engineering, Graduate School of System Design
Tokyo Metropolitan University
Tokyo, Japan
fernando-ardilla@ed.tmu.ac.jp, aa.saputra@tmu.ac.jp, kubota@tmu.ac.jp

2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-5258-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/SMC53654.2022.9945096

***Abstract*—This papers describes a learning algorithm for growing neural gas to construct a topology-preserving map from a 3D point cloud whose topology can change dynamically. Growing Neural Gas with Utility Factor (GNG-U) has been presented as a method for learning the topology of a 3D space environment and applying it to non-stationary or dynamic data distribution. However, when a node is added to an existing network after several errors with sampling data have accumulated, it is difficult for a standard GNG-U to considerably boost learning speed. As a result, we propose a revolutionary growth strategy that dramatically accelerates learning and convergence. This method immediately adds a sample of data as a new node to an existing network based on the likelihood of node addition estimated by the distance to the third closest node and the first and second closest nodes at maximum. Experiment findings show that the proposed algorithm's network can quickly adapt to represent the topology of non-stationary input distributions.**

***Index Terms*—Topological structure, Growing Neural Gas, Batch learning.**

I. Introduction

Many unsupervised learning approaches [1], [2] have been applied to data mining tasks. For example, we need an online real-time processing system for 3D distance data for self-driving cars. Extraction of topological information from 3D distance data improves processing efficiency. Unsupervised learning aims to locate essential information quickly via extracting features, grouping, and topological mapping [3]. Furthermore, several approaches have been developed, and proven efficacy [4]. However, there are still several issues with unsupervised real-time learning. The number of data sampling times required for clustering and topological mapping is generally significant. To substantially minimize the computational cost and data sampling times, we focus on two critical unsupervised learning targets in this study. One is rapid real-time clustering, and the other is topological mapping efficiency. We also strive to limit the number of hyper parameters required to optimize a dataset's quality and quantity. The parameters are challenging to set since they often depend on a data structure [5].

There are three primary methods of learning: stochastic gradient descent, mini-batch gradient descent, and entire batch gradient descent. Standard GNGs employ an iterative stochastic gradient descent method online, and learning rates do not depend on iteration count. So, to promote learning convergence, GWR was devised [6]. For the purpose of computing the weight update, the mini-batch gradient descent uses more than two data points while remaining smaller than the overall size. The mini-batch gradient descent algorithm is often used in deep learning but is not used in clustering approaches.

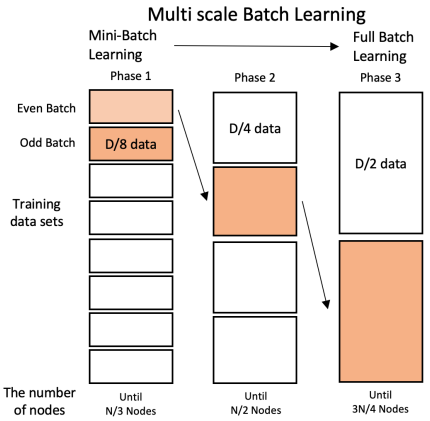


Figure 1. Learning phase in Multi-Scale Batch Learning-GNG of dynamic data. $\lambda_L = \{D/8, D/4, D/2\}$; D are the number of data, $\mu_L = \{N/3, N/2, 3N/4\}$; N are the maximal number of nodes.

The original Batch Learning GNG (BL-GNG) was proposed to improve the convergence property of GNG based on entire batch learning in the final stage [7] Based on those above, we presented a multi-scale batch-learning technique for GNG (MSBL-GNG) [8], [9]. However, ordinary GNG, GWR, and BL-GNG cannot significantly speed up learning because a node is added to an existing network after error accumulated from sampling data several times. So, in this study, we propose a novel growing method for MSBL-GNG. This method adds a new node to an existing network based on the distance between the third nearest node and the first and second nearest nodes. The paper's main contribution is to accelerate the GNG algorithm's learning convergence on dynamic data distribution.

II. Related Work

Many topological clustering algorithms have been proposed. The learning algorithm uses online incremental unsupervised learning (GCS) [10], GNG. As previously stated, GNG and GCS can alter the topological structure based on the adjacent relation connecting to the ignition frequency of a nearby node. However, GNG may eliminate nodes, and edges from the topological structure using the ages of each edge [10]. Hasegawa and Shen presented a Self-Organizing Incremental Neural Network (SOINN) [11] to improve GNG's learning capabilities. SOINN tolerates noise well and clusters well. Load-Balancing SOINN (LB-SOINN) enhances learning stability and clustering performance [12] by balancing the learning time of each node. However, LB-SOINN requires six user-defined

parameters, which are challenging to optimize. FCM-BL-GNG was proposed to reduce user-defined parameters and learning convergence [7], [13], [14]. Due to the batch gradient descent, the FCM-BL-GNG learning pace is slower than standard online GNG.

The GNG technique is most effective when dealing with stationary or slowly moving distributions. GNG has been used for various application, such as hand recognition [15]. In the previous research, we have applied the GNG in for several robotics applications, such as vertical ladder detection [16], object's grasp position recognition [17], and locomotion modelling [18], [19], and map building [13]. Fritzke's [20] proposed Growing Neural Gas with Utility (GNG-U) is a competitive learning approach that dynamically changes the topological structure of the network based on the edge referring to the ignition frequency of the adjacent node in response to the accumulated error. The GNG-U algorithm manages these eventualities by moving no longer useful nodes. In essence, GNG-U eliminates nodes that contribute little to error reduction in favor of placing them where they can make a more significant contribution. However, a carelessly chosen utility factor affects the algorithm's behavior, and the unstable position of the node will cause problems if it is used for recognition. For this reason, we use MSBL-GNG, which has the advantage of speed in producing convergent output data. Also, the stability of the nodes can be known quickly, namely at the end of the complete batch learning process, with the number of nodes approaching the maximum number of nodes.

III. Proposed method

This subsection discusses a multi-scale data update technique in MSBL-GNG that enables quick and efficient learning for dynamic data. Generally, a training approach called stochastic gradient descent, mini-batch gradient descent, or batch gradient descent is used. The original GNG uses an iterative learning method of stochastic gradient descent in an online way.

Figure.1 illustrates a multi-scale batch learning (MSBL) update technique with N and D as the number of nodes and data points, respectively. A dataset's total size is divided into eight mini-batch datasets, with the quantity of data in the learning phase (L) denoted by λ_L. The individual mini-batch dataset is used sequentially to update nodes in GNG.

First, we discuss how to transition to the next phase to deal with dynamic data over time. Different from previous work [8], [9], the first mini-batch learning is done when the following dataset is given to the network. Therefore, we should restart the learning from the second initialization to the next phase due to the initial screening. After that, the phase transition is done one by one, as shown in Fig.1. Finally, the entire batch learning will be conducted at least once to update reference vectors and their edge connectivity. As a result, we need a little less than D of data sampling, at least exactly 7/8D.

$$D_{sum} = \sum_{i=2}^{L_{max}} \frac{1}{2^{i-1}} D < D \tag{1}$$

A. Overview of Multi-Scale Batch Learning GNG

The main notations used in MSBL-GNG are shown in Table.I, and we explain a learning algorithm of MSBL-GNG as follows:

Table I
Notation used in MSBL-GNG

Symbol	Description
w_i	the n-th dimensional vector of i-th node.
A	a current set of nodes
E_i	accumulated error variable
C_i	a set of nodes connected to the i-th node
x_i^1	selection times of the i-th node
x_i^2	selection times of the i-th node selected for the topological learning
$c_{i,j}$	edge between i-th and j-th nodes
$c_{i,j}^{'}$	selection times of the edge between the i-th and j-th nodes.
$p_k(v)$	the probability of adding a sample data to the network based on AIS-based addition strategy

1) (Initialization) Create three connected nodes at their edges. Initialize the learning phase level ($L = 1$). The sample data is sorted randomly. Set the initial sampling data ID (it) to 1. The growing phase is 1 ($k = 1$).
2) (Data Update) Update sample data.
3) (Mini-batch Initialization) Initialize the temporal weight update ($\Delta w_i = 0$), selection times ($x_i^1 = 0, x_i^2 = 0$) and the temporal edge connectivity ($c_{i,j}^{'} = 0$), $\forall_{i,j} \in A$.
4) (Initial Screening) Restart the initialization by using the current dataset if the rate of nodes in the first mini-batch to the previous tie step is the predefined threshold.
5) (Update Δweight) According to a sample of data, update the temporal weights and the temporal edge connection.
   a) Select the first nearest node, s_1 and the second nearest unit, s_2, (and third nearest node, s_3 after L_2 as well) according to the distance, $d_i = \|v - w_i\|$ with a sample data v,
   $$s_1 = \underset{i \in A}{\operatorname{argmin}} d_i \tag{2}$$
   $$s_2 = \underset{i \in A \setminus (s_1)}{\operatorname{argmin}} d_i \tag{3}$$
   $$s_3 = \underset{i \in A \setminus (s_1, s_2)}{\operatorname{argmin}} d_i \tag{4}$$
   b) Add the distance between the input and reference vectors to the accumulated error.
   $$E_{s_1} \leftarrow E_{s_1} + \eta_1 d_{s_1} i \tag{5}$$
   c) Calculate the winner's and direct topological neighbors' weight updates.
   $$\Delta w_{s_1} \leftarrow \Delta w_{s_1} + \eta_1 (v - w_{s_1}) \tag{6}$$
   $$\Delta w_j \leftarrow \Delta w_j + \eta_2 (v - w_j) \quad if c_{s_1,j} = 1 \tag{7}$$
   where η_1 and η_2 are the learning rates. Increment the selection times $x_{s_1}^1 ++$, $x_{s_1}^2 ++ (j \in c_{s_1})$. Increment the temporal edge connectivity ($c_{s_1,s_2}^{'} ++$).
6) (Add If Silent - based Node Addition) Add a new node to the network with the probability of $pL(v)$ for each sample data v.
7) Increment the sampling data ID,it. Move ahead with Step 3 if the number of iterations is not an integer multiple of μ_L.
8) (Mini-batch Update) Update the weights by the MSBL, update the edge connectivity, and remove the nodes with $x_i = 0$.
   $$\Delta w_i \leftarrow w_i + \Delta w_i / (x_i^1 + x_i^2), \quad if (x_i^1 + x_i^2) > 0 \tag{8}$$

$$c_{i,j} = \begin{cases} 1 & if \quad c'_{i,j} \geq 1 \\ 0 & otherwise \end{cases} \tag{9}$$

9) (Node Deletion) Delete a node according to the selection-based deletion strategy.
10) (Error-based Node Addition) Insert new nodes according to the error-based addition strategy.
11) If the number of nodes does not meet the preset value μ_L, go to step 2.
12) (Phase Transition) Increase the level of the learning phase, $L(L < L_{max})$ and the growing phase level, $k(k \leq 3)$, and move it to the top of the next mini-batch data set.
13) (Full-batch Learning) Conduct a full batch learning.
14) If the stopping criterion (e.g., network size or performance) is not met, go to step 2.

MSBL-GNG and the original GNG age differently. The mini-batch dataset refreshes edge connections, deleting non-selected edges and nodes without edges. MSBL-GNG doesn't employ age parameters in batch learning and updates learning rates and reference vectors. MSBL-GNG applies a higher learning rate to the second closest node in the beginning to cover the data dispersion. This approach slows with many nodes.

$$\eta_2 = \alpha_2.(L_{max} - L) \tag{10}$$

$$\eta_1 = 1.0 - \alpha_2.(L_{max} - L) \tag{11}$$

where L_{max} is the maximal learning phase level, and α is a coefficient. The learning rate to the nearest node is increasing toward the full batch learning as follows;

Finally, the full batch learning is conducted in the last learning phase $(L = L_{max})$ to update reference vectors and their edge connectivity.

B. Nearest neighbor search optimization

In high-dimensional data, learning MSBL-GNG is time-consuming. Aside from the complexity analysis, the nearest neighbor search takes up roughly 67% of the algorithm runtime. So we recommend combining our MSBL-GNG extension for point cloud sequences with other optimization strategies described in [21], [22]. We recommend a growing one since we employ this optimization to modify the initial map's topology to a sequence of point clouds. The uniform grid has some performance difficulties. These fundamental difficulties pertain to voxel and grid size selection. The block of the input data is used to choose grid dimensions. A block is constructed using the input data's minimum and maximum 3D points. The voxel size is determined by the mean point cloud resolution versus the number of neurons or greater than c. in our case $c = 0.075$. In [21] has more information about this optimization step. Figure 2 illustrates a 3D grid that can be used to speed up the learning step of MSBLGNG.

C. Growing Procedure

This subsection explains how to add nodes to the network and delete nodes from the network. Fig.3a shows an example of topological clustering. In general, n+1 points are required to generate a closed area in the n-dimensional space. If a cluster should be composed of three nodes at least for the aim, the distance information to these three nodes should be used in each cluster. The original GNG uses only the first and second nearest nodes to connect an edge between two nodes.

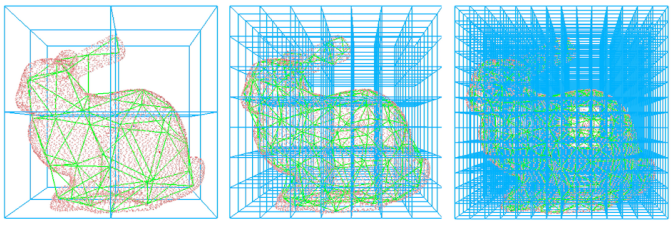


Figure 2. Growing uniform grid used to accelerate the MSBL-GNG algorithm

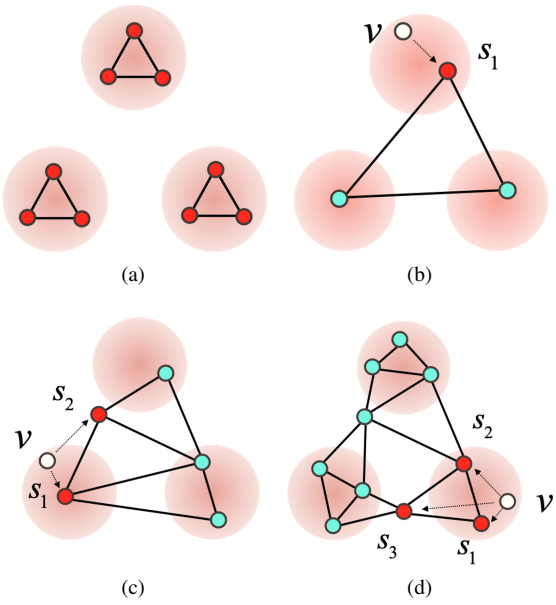


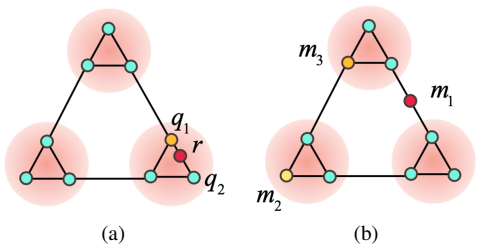
Figure 3. AIS-based addition strategy for topological clustering. (a) Topological clustering. (b)1st phase level(L=1). (c)2nd phase level(L=2). (d)3rd phase level(L=3).

In contrast, MSBL-GNG uses the third nearest node and them. We divide the growing procedure into three phases corresponding partially to the multi-scale phase level. In the first growth phase, MSBL-GNG tries to cover a given dataset roughly by using the first nearest node (Fig.3b). In the second growth phase, MSBL-GNG tries to make clusters like the original GNG (Fig.3c). In the 3rd or larger phase level, MSBL-GNG tries to conduct topological clustering with three nodes at least in the cluster (Fig.3d). To realize the above growing procedure, MSBL-GNG adds sample data to the network with the probability $p_k(v)$ of as follows;

$$p_k(v) = max(0, \tanh(z_k(v)/l_{range})) \tag{12}$$

$$z_k(v) = \sum_{i=1}^{k} \gamma_{k,i} d_{s_i}, l_{range} = l_{Max} - l_{Min} + \varepsilon \tag{13}$$

whare k is the growing phase level (k = 1, 2, 3), $\gamma_{k,i}$ is the weight parameter of the i-th nearest node in the k-th growing phase, l_{Max}, l_{Min} and l_{Ave} are the maximal, minimal, and average distance between two connected nodes, $c_{i,j} = 1(i, j \in A)$, respectively, l_{range} is the range of the distance information l_{Max} and l_{Min}, l_k is l_{Min}, l_{Ave}, or l_{Max} according to the aim of topological clustering such as the learning speed and clustering accuracy, and ε is the small positive constant ($\varepsilon > 0$). In this paper, we use $l_1 = l_{min}$, $l_2 = l_{ave}$ and $l_3 = l_{ave}$. If the condition is satisfied, the sample data v to the network, $w_r \leftarrow v$ where r



(a) (b)

Figure 4. (a).Error-based addition strategy: A node is added to the network with the rate of $E_{q1} : E_{q2}$. (b).Selection-based deletion strategy: A node is removed from the network according to the selection times of the nearest node

is the new node ID, and the edges connecting the new node r with nodes s_1 and s_2 are added, and the original edge between q_1 and q_2 is removed ($c_{r,s_1} = 1$, $c_{r,s_2} = 1$, and $c_{s_1,s_2} = 0$).

MSBL-GNG also uses an error-based addition strategy like the original GNG. Still, we improve the growing performance by using multiple times of additions and rate-based insertion (Fig.3). We explain the error-based addition strategy of Step 10 in the following.

10-1 Select the node q_1 with the maximum accumulated error and the node q_2 with the maximum accumulated error among the neighbors of q_1.

$$q_1 = \underset{i \in A}{\operatorname{argmax}} E_i, \quad q_2 = \underset{i \in C_{q1}}{\operatorname{argmax}} E_i, \tag{14}$$

10-2 Add a new node r to the network and use the rate of E_{q1} : E_{q2} to interpolate its reference vector between q_1 and q_2.

$$w_r = \frac{1}{E_{all}} \left(E_{q_1} . w_{q_1} + E_{q_2} . w_{q_2} \right), E_{all} = E_{q_1} + E_{q_2} \tag{15}$$

10-3 Replace the existing edge between q_1 and q_2 with the edges linking the new node r to the nodes q_1 and q_2 ($c_{r,q_1} = 1$, $c_{r,q_2} = 1$, $c_{q_1,q_2} = 0$).

10-4 Decrease the error variables of q_1 and q_2,

$$E_{q_1} \leftarrow E_{q_1} - \frac{E_{q_1}^2}{2E_{all}}, E_{q_2} \leftarrow E_{q_2} - \frac{E_{q_2}^2}{2E_{all}} \tag{16}$$

10-5 Interpolate the error variable of r from q_1 and q_2,

$$E_r \leftarrow \frac{1}{2E_{all}} \left(E_{q_1}^2 + E_{q_2}^2 \right) \tag{17}$$

Furthermore, multiple node additions are used in proportion to the number of learning phase levels L. We use a small number of additional times in the beginning because the number of nodes and samplings is insufficient. We can increase the node addition times as the learning phase increases. An efficient number of weight updates is required to obtain optimal topological structure when a node is added to a network. An unnecessary node may be added to the network if many nodes are added to the network in the short iterations by the above addition strategies. Therefore, we propose a selection-based deletion strategy of Step 9 (Fig.4b).

9-1 Select three nodes selected rarely,

$$m_1 = \underset{i \in A}{\operatorname{argmin}} x_i, \tag{18}$$

$$m_2 = \underset{i \in A \setminus (m_1)}{\operatorname{argmin}} x_i, \tag{19}$$

$$m_3 = \underset{i \in A \setminus (m_1, m_2)}{\operatorname{argmin}} x_i, \tag{20}$$

9-2 Remove the node m_1 from the network ($c_{m_1,i} = 0 (i \in C_i)$) if this condition is satisfied.

$$if \quad x_{m_1}^1 \leq \tau_1 x_{m_2}^1 + \tau_2 x_{m_3}^1 \tag{21}$$

9-3 As a result of node deletion, remove the nodes having no edge.

IV. Experimental Results

A. Experiment Conditions

This section shows some experimental results to demonstrate the performance of our proposed method. Experiments were conducted on a computer with 1.6 GHz Intel Core i5 and 8GB RAM. In the first experiment, we will investigate the proposed method for tracing 3D objects through a sequence's input space. The suggested method can then estimate the temporal trajectory of a 3D non-stationary distribution. Open dynamics engine(ODE) simulated 3D non-stationary distributions. ODE allowed us to create synthetic scenes and see them like a 3D Time-of-Flight camera. This software's key benefit is providing ground truth (no noise) data. It also captures point cloud sequences emulating camera movement or object movement. On the other hand, it offers the distribution's translation and rotation values for each series frame.

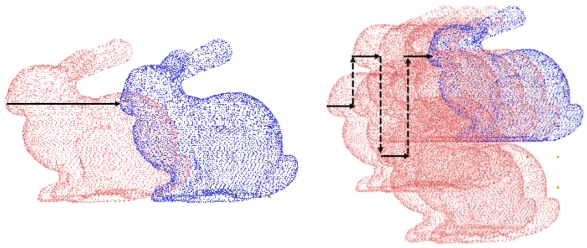
Regarding a parameter of MSBL-GNG, Coefficients of used for AIS-based addition strategy in the growing phase level where the coefficients are set according to the rate of $\gamma_{2,1}$: $\gamma_{2,2} = 2:1$ and $\gamma_{3,1} : \gamma_{3,2} : \gamma_{3,3} = 3:2:1$. The coefficients used for selection-based node deletion is τ_1 = 1.0 and τ_2 = 0.2 .The learning rates are α_1 = 0.03 and α_2 = 0.05 . The threshold numbers of nodes used for the learning phase transition are $\mu_1 = N/10$, $\mu_2 = N/3$, $\mu_3 = N/2$, and $\mu_L = N.L/(L+1)$ if $(L > 3)$. We experimentally decided on these parameters, but we used the same parameters in comparing benchmark tests to show the generality of the above parameter setting. The number of the maximal learning phase levels, L_{max}, only depends on the number of data, D, and the maximal number of nodes, N. The parameters set up in the experiment include $L_{max} = 8$, from the 3D Stanford Bunny data set has 17974 points (D=17974) and maximum node GNG has 1000 nodes (N=1000).

B. Tracking error

On the right is the input adaptation error along some point cloud sequences. It is important to know the distance between the generated structure and the input data since self-organizing neural networks are generally quantified in terms of topological preservation. This metric compares our produced model to the original input data. To quantify the resulting map's input space adaption, we computed its Mean Square Error (MSE) against sampled points (input space).

$$\text{MSE} = \frac{1}{|A|} \sum_{\forall v \in A} \min_{i \in A} d|v - w_i| \tag{22}$$

where A is the input space, v is a sample from the input space, i is the neuron with the smallest distance to the sample from the input space, and w_i is the neuron's weight. $d|v - w_i|$ is the euclidean distance between the nearest neuron and the input point v. Once the network topology learning phase is complete, this measurement is computed. Figure 6 shows



(a) (b)

Figure 5. Point cloud sequence Bunny dataset (a). Moving along the X-axis (b). Moving X-axis and Y-axis

Figure 6. MSBL-GNG Representation Mean Square Error and Maximum Error).

the MSE response and the maximum error generated in the sequential data shown in Figure 5. The response shows the convergence speed of the nodes generated from the MSBL-GNG method. The MSE value of the two experiments showed below 0.04. A sizeable maximum error change occurs if the input data changes are significant.

C. Optimized MSBL-GNG algorithm

Some tests were carried out to verify the optimized implementation of the MSBL-GNG algorithm. To further demonstrate the benefits of our implementation, we conducted performance tests to compare the increase in speed achieved by our method with an alternative (Original MSBL-GNG). Our improved version are shown in Table II for updating an initial map for non-stationary input data. With a consistent grid layout, we can see how much less time it takes the network to update the map to the current frame.

D. Experiment in the real environment with ROS(Robot Operating System)

To show the effectiveness and applicability of the proposed model, we also experimented with MSBL-GNG in a real environment. The experiments used point cloud input from an

Table II
EVALUATED EXECUTION TIMES PER FRAME FOR MSBL-GNG WITH AND WITHOUT UNIFORM GRID

	without UG	with UG
N=500	0.53s	0.31s
N=1000	1.09s	0.53s
N=1500	1.74s	0.77s
N=2000	2.43s	0.93s

RGB-D camera with more than 150,000 points. As MSBL-GNG input, we randomly take as much as 10,000 data for each input sampling. The setting parameters include $L_{max} = 8$, D=10000, N=300, $L_{t=0} = L_{max}$, $L_{t+1} = L_{max} - 6$. This experiment builds a topological map with GNG, which will detect objects on the table. The object on the table is moved randomly to see the response of the topology map. Fig.7 shows the results of the topology map running on ROS and visualized on rviz. The green lines are the result of the topology map. Fig.8 shows the MSBL-GNG response when the object in front of it moves randomly. The topology map will converge or flatten when the number of nodes has reached the maximum. On rostopic starting from subscribing to point cloud from RGB-D to publishing nodes and edges of GNG. This experiment with a value of $L = L_{max} - 6$ ROS node can publish up to 10Hz.

V. CONCLUSION

In this paper, we present Multi-Scale Batch Learning Growing Neural Gas (MSBL-GNG) for improving the learning stability of GNG on dynamic data distribution. Multi-Scale Batch Learning-GNG (MSBL-GNG) to study the topological structure of various dataset scales. Several experiments have been carried out to evaluate our proposed method. The proposed approach uses a grid layout to reduce computing complexity and provide quicker acceleration factors than the original MSBL-GNG algorithm. The computational cost grows as the volume of input data and the maximum number of nodes increase. However, it can be optimized by defining the optimum value of L after the first cycle. The proposed model is effectively applied for batch learning applications. It has been validated in real-world applications using ROS. The fasting can attentively reconstruct the topological structure when the object is moved randomly. We will address limitations and implement them on robots such as obstacle detection and 3D object segmentation in future work.

ACKNOWLEDGMENT

This work was partially supported by JST [Moonshot RnD][Grant Number JPMJMS2034].

REFERENCES

[1] D. Shi, L. Zhu, Y. Li, J. Li, and X. Nie, "Robust structured graph clustering," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 11, pp. 4424–4436, 2019.
[2] Z. Uykan, "Fusion of centroid-based clustering with graph clustering: An expectation-maximization-based hybrid clustering," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[3] P. E. H. R. O. Duda and D. G. Stork, *Pattern classification*. 2nd ed, John Wiley Sons, 2012.
[4] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang, "Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering," *IEEE Transactions on Neural Networks*, vol. 22, no. 11, pp. 1796–1808, 2011.

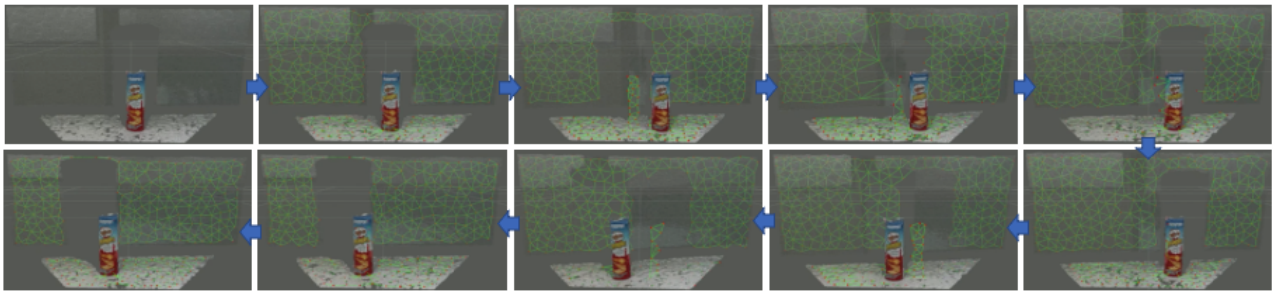


Figure 7. Visualization MSBL-GNG on Rviz with moving object

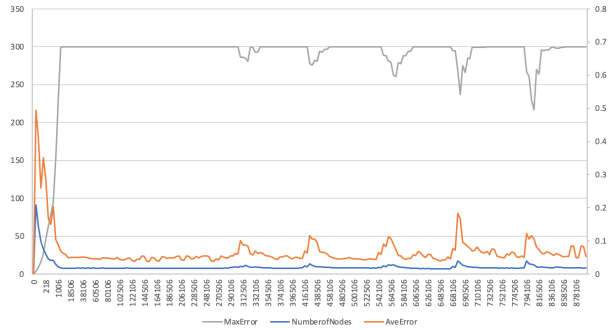


Figure 8. MSBL-GNG with input dataset B($L = 4$) on ROS Environment

[5] D. Heinke and F. H. Hamker, "Comparing neural networks: A benchmark on growing neural gas, growing cell structures, and fuzzy artmap," *IEEE transactions on neural networks*, vol. 9, no. 6, pp. 1279–1291, 1998.

[6] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural networks*, vol. 15, no. 8-9, pp. 1041–1058, 2002.

[7] Y. Toda, T. Matsuno, and M. Minami, "Multilayer batch learning growing neural gas for learning multiscale topologies," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 25, no. 6, pp. 1011–1023, 2021.

[8] M. Iwasa, N. Kubota, and Y. Toda, "Multi-scale batch-learning growing neural gas for topological feature extraction in navigation of mobility support robots," in *The 7th International Workshop on Advanced Computational Intelligence and Intelligent Informatics (IWACIII 2021)*, 2021.

[9] N. Doteguchi and N. Kubota, "Topological tracking for mobility support robots based on multi-scale batch-learning growing neural gas," in *In Proc. of 10th International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, 2021.

[10] B. Fritzke, "Growing cell structures—a self-organizing network for unsupervised and supervised learning," *Neural networks*, vol. 7, no. 9, pp. 1441–1460, 1994.

[11] S. Furao and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," *Neural networks*, vol. 19, no. 1, pp. 90–106, 2006.

[12] H. Zhang, X. Xiao, and O. Hasegawa, "A load-balancing self-organizing incremental neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1096–1105, 2013.

[13] Y. Toda, W. Chin, and N. Kubota, "Unsupervised neural network based topological learning from point clouds for map building," in *2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*. IEEE, 2017, pp. 1–6.

[14] Y. Toda and N. Kubota, "Topological structure learning based enclosing formation behavior for monitoring system," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 831–836.

[15] N. Mirehi, M. Tahmasbi, and A. T. Targhi, "Hand gesture recognition using topological features," *Multimedia Tools and Applications*, vol. 78, no. 10, pp. 13 361–13 386, 2019.

[16] A. A. Saputra, W. H. Chin, Y. Toda, N. Takesue, and N. Kubota, "Dynamic density topological structure generation for real-time ladder affordance detection," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 3439–3444.

[17] A. A. Saputra, C. W. Hong, and N. Kubota, "Real-time grasp affordance detection of unknown object for robot-human interaction," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 3093–3098.

[18] A. A. Saputra, J. Botzheim, A. J. Ijspeert, and N. Kubota, "Combining reflexes and external sensory information in a neuromusculoskeletal model to control a quadruped robot," *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.

[19] A. A. Saputra, N. Takesue, K. Wada, A. J. Ijspeert, and N. Kubota, "Aquro: A cat-like adaptive quadruped robot with novel bio-inspired capabilities," *Frontiers in Robotics and AI*, vol. 8, 2021.

[20] B. Fritzke, "A self-organizing network that can follow non-stationary distributions," in *International conference on artificial neural networks*. Springer, 1997, pp. 613–618.

[21] D. Fišer, J. Faigl, and M. Kulich, "Growing neural gas efficiently," *Neurocomputing*, vol. 104, pp. 72–82, 2013.

[22] S. Orts-Escolano, J. Garcia-Rodriguez, V. Morell, M. Cazorla, M. Saval, and J. Azorin, "Processing point cloud sequences with growing neural gas," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–8.