# Topological Mapping for Event-based camera using Fast-GNG and SNN

Naoki Doteguchi
*dept. Mechanical Systems Engineering*
Tokyo Metropolitan University
Tokyo, Japan
doteguchi-naoki@ed.tmu.ac.jp

Naoyuki Kubota
*dept. Mechanical Systems Engineering*
Tokyo Metropolitan University
Tokyo, Japan
kubota@tmu.ac.jp

***Abstract*—In recent years, research has been conducted on digital transformation toward the realization of Society 5.0. Among this research, there has been little research on monitoring that takes advantage of the features of 5G communications. To realize such ultra-real-time monitoring, it is necessary that predict and respond to the movement of the measurement target without large-scale computer simulations. Therefore, we focused on Event-based cameras. Unlike conventional cameras, event-based cameras are inspired by living organisms in which each pixel outputs luminance changes asynchronously and have excellent features such as high time resolution, high dynamic range, low latency, and low power consumption. These features have attracted attention in the fields of robot vision and computer vision, and the system is a compatible solution to realize such monitoring. We have also used topological mapping techniques to deal with various problems, and we are convinced that such an approach will also be effective for Event-based cameras. However, a lot of noise must be dealt with when trying to learn topological structures using Growing Neural Gas (GNG). However, Event-based cameras are, in principle, completely different from conventional cameras and require a novel methodology. Event-based cameras work well with Spiking Neural Networks (SNN) because of their principle. Therefore, to realize ultra-real-time monitoring, this paper proposes a model to obtain the topological structure of an Event-based camera with noise reduction by using SNN and GNG.**

***Keywords—Event-based camera, Topological Map, Spiking Neural Networks, Growing Neural Gas***

## I. Introduction

In recent years, research and development aimed at digital transformation have been conducted toward realizing Society 5.0. Especially in research and development using 5G Communications, there has been little comprehensive research on monitoring that takes advantage of the features of 5G. To conduct such 5G monitoring, measuring most effectively after predicting the data needed next time based on the analysis results from the measured data up to a certain time is important. Measuring and tracking objects in a situation-dependent and active method is necessary to achieve these goals. This requires measurement while controlling the attitude of the sensor and prediction based on dynamics regarding the physical behavior using the position, attitude, and geometry of the tracked object. In addition, estimating and predicting the physical behavior of the target to be tracked and the phenomena to be targeted often requires large-scale computer simulations, which require huge computational costs. Such predictions are, therefore, necessary for ultra-real time monitoring without requiring extensive computer simulations. An Event-based camera is a visual sensor inspired by biological organisms in which each pixel outputs a luminance change asynchronously, in contrast to a normal camera, which outputs an image frame by frame. Event-based cameras have outstanding features not found in traditional cameras, such as high time resolution, high dynamic range, low latency, and low power consumption. These features have attracted attention in the fields of robot vision and computer vision, which are also useful for realizing ultra-real-time monitoring. Also, feature extraction must be done appropriately to make predictions without extensive computer simulation. We have proposed feature extraction, prediction, and simulation applications, using topological maps and graph structures learned by GNG for various objects [1, 2]. Such topological mapping methods are effective for various problems, and a similar approach should be possible for Event-based cameras. However, there is a lot of noise in the output of Event-based cameras, which is a very serious problem for learning GNG in Event-based cameras with polarity output. A method such as taking the average in the neighborhood is considered for traditional camera images to remove such noise. However, a new methodology is needed for Event-based cameras because they are completely different from traditional cameras in principle. General noise reduction filters applied to camera images are adapted to those that are two-dimensional information as images.

On the other hand, the output of an Event-based camera is the pixel coordinates where the luminance change occurred, event polarity, and the time it occurred, so it cannot be treated like a normal image. There is also the problem of increased data volume and loss of features unique to Event-based cameras when conventional methods are applied to 2D images. From this perspective, a noise reduction methodology adapted to Event-based cameras is needed. Here, it is known that Event-based cameras are compatible with Spiking Neural Networks (SNN) due to the principle of outputting event polarity. SNN is also highly resistant to noise due to its internal state its characteristics. In this paper, we propose a model of SNN corresponding to asynchronous event-based camera input to eliminate noise while taking advantage of the asynchronous and high frame rate features of Event-based cameras. Then, the topological structure is obtained using GNG, which enables feature extraction for ultra-real-time monitoring.

## II. Topological Mapping for Event-based camera

### A. Spiking Neural Networks (SNN)

Artificial neural networks can be divided into pulse neuron models and firing frequency neuron models [3]. The firing frequency neuron model has been widely studied in the field of artificial neural networks. This model assumes that a time average of spikes represents information and that each spike

2022 IEEE Symposium Series on Computational Intelligence (SSCI) | 978-1-6654-8768-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/SSCI51031.2022.10022102

is insignificant. Such features dominate at the ends of the motor and sensory system, including the magnitude of sensory stimuli and muscle control signals. In contrast, the pulse neuron model directly describes and models spikes, so it is possible to construct a model in which the behavior of the network changes depending on the presence or absence of a single spike or the timing of a single spike's firing. The most important feature of the pulse neuron model is that it can learn not only spatial patterns but also time-series patterns because it processes information according to the time interval of the spike sequence [4, 5]. In addition, the principle of the device is that it has an internal state, which makes it highly resistant to noise. In this study, a simplified pulse neuron model, the Spiking Neural Network (SNN), is applied to Event-based camera noise reduction. However, it is necessary to extend and apply the SNN model to accommodate asynchronous input and output, a feature of Event-based cameras. In the normal SNN, the internal state is updated step by step, and the timing and frequency of internal state updates are the same for all nodes, but this needs to be changed to accommodate asynchronous inputs and outputs. Therefore, to deal with asynchronous updates, the proposed method stores the time $T_{ij}$ when the node fired and the time $T_{ij}^{in}$ when the last input occurred and determined the amount of update using the difference between the time $t$ when the input occurred and the time $T_{ij}, T_{ij}^{in}$. This allows each node to be treated in a unified way for asynchronous data. The equation for the model in this study is as follows.

Since the input of an Event-based camera has binary values (positive and negative), the SNN can handle this by putting two nodes at each pixel. First, the internal state of each node can be calculated as $h_{ij}(t)$ following (1), (2). The superscripts + and - correspond to positive and negative input.

$$h^{+}{}_{ij}(t) = h^{sys+}{}_{ij}(t) + h_{ij}^{ref+}(t) + w \cdot q^{+}{}_{ij}(t) \quad (1)$$

$$h^{-}{}_{ij}(t) = h^{sys-}{}_{ij}(t) + h_{ij}^{ref-}(t) + w \cdot q^{-}{}_{ij}(t) \quad (2)$$

Where $h^{sys}{}_{ij}(t)$ is the input from each neuron to neuron *(i, j)*, and$h_{ij}^{ref}(t)$ is the refractory period of the neuron. $q_{ij}(t)$ is the input from the external environment. Also, $w$ is a weight. When the internal state of a neuron exceeds a threshold value, the neuron fires. If the threshold value is $\theta$, it can be expressed as (3), (4).

$$P^{+}{}_{ij}(t) = \begin{cases} 1 & h^{+}{}_{ij}(t) \geq \theta \\ 0 & otherwise \end{cases} \quad (3)$$

$$P^{-}{}_{ij}(t) = \begin{cases} 1 & h^{-}{}_{ij}(t) \geq \theta \\ 0 & otherwise \end{cases} \quad (4)$$

It is also calculated by (5) and (6) for $h^{sys}{}_{ij}(t)$.

$$h^{sys+}{}_{ij}(t) = \gamma_1{}^{\left(t-T_{ij}^{+}\right)} \cdot h^{+}{}_{ij}\left(T_{ij}^{in+}\right) \quad (5)$$

$$h^{sys-}{}_{ij}(t) = \gamma_1{}^{\left(t-T_{ij}^{-}\right)} \cdot h^{-}{}_{ij}\left(T_{ij}^{in-}\right) \quad (6)$$

In the proposed model, the input relationship from other neurons is not defined, so the internal state is attenuated using the attenuation rate $\gamma_1{}^{(t-T_{ij})}$ for the internal state. where $\gamma_1{}^{(t-T_{ij})}$ is calculated using the current time t and $T_{ij}$.

It is then calculated using (7), (8) for $h_{ij}^{ref}(t)$.

$$h_{ij}^{ref+}(t) = \begin{cases} \gamma_2{}^{\left(t-T_{ij}^{in+}\right)} \cdot h_{ij}^{ref+}\left(T_{ij}^{in+}\right) - R \\ \qquad if\ P^{+}{}_{ij}(\tau) = 1\ \left(T_{ij}^{+} < \tau \leq t\right) \\ \gamma_2{}^{\left(t-T_{ij}^{in+}\right)} \cdot h_{ij}^{ref+}\left(T_{ij}^{in+}\right) \\ \qquad otherwise \end{cases} \quad (7)$$

$$h_{ij}^{ref-}(t) = \begin{cases} \gamma_2{}^{\left(t-T_{ij}^{in-}\right)} \cdot h_{ij}^{ref+}\left(T_{ij}^{in-}\right) - R \\ \qquad if\ P^{-}{}_{ij}(\tau) = 1\ \left(T_{ij}^{-} < \tau \leq t\right) \\ \gamma_2{}^{\left(t-T_{ij}^{in-}\right)} \cdot h_{ij}^{ref-}\left(T_{ij}^{in-}\right) \\ \qquad otherwise \end{cases} \quad (8)$$

where R>0 and suppresses the neuron that has fired for a certain time. $\gamma_2{}^{(t-T_{ij})}$ is the decay rate calculated using the current time t and $T_{ij}$. The input from the external environment, $q_{ij}(t)$, is calculated in the proposed model as (9) and (10) using the inputs of the corresponding pixel and its surrounding pixels in the SNN.

$$q^{+}{}_{ij}(t) = input_{ij}^{+}(t) + \sum_{m} \{\gamma_3{}^{(t-T_m^{in+})} \cdot input_m^{+}(T_m^{in+}) - \gamma_3{}^{(t-T_m^{in-})} \cdot input_m^{-}(T_m^{in-})\} \quad (9)$$

$$q^{-}{}_{ij}(t) = input_{ij}^{-}(t) + \sum_{m} \{(\gamma_3{}^{(t-T_m^{in-})} \cdot input_m^{-}(T_m^{in-}) - \gamma_3{}^{(t-T_m^{in+})} \cdot input_m^{+}(T_m^{in+})\} \quad (10)$$

$\gamma_3{}^{(t-T_{ij})}$ is the decay rate calculated using the current time t and $T_{ij}$ and $input_{ij}(t)$is the pixel input and {-1,1}. Also, m represents the neighborhood relation.

### *B. Growing Neural Gas (GNG)*

Various types of Growing Neural Gas (GNG) have been proposed for learning topological structures [6~9]. GNG was originally proposed by Fritzke as one of the unsupervised learning neural networks such as Self-Organizing map (SOM) [10], Neural Gas (NG) [8], and Growing Cell Structure (GCS) [9]. In SOM, the number of nodes and topological structure of the network is predetermined, whereas, in NG, the number of nodes is fixed in advance. However, the topological structure is learned according to input data distribution. On the other hand, GCS and GNG can dynamically change their topological structure by adding edges and nodes using the error. But GCS does not delete nodes or edges, whereas GNG can delete nodes and edges based on the concept of age. And these methods have been successfully applied to various real-world problems [11~14]. To improve learning performance, various types of GNG have been proposed. The Batch Learning-GNG (BL-GNG) was proposed to improve the convergence of the original GNG [15], and the Multi-scale BL-GNG (MS-BL-GNG) method was proposed to improve the learning speed by changing the size of the mini-batch used for training according to the learning phase [16]. In addition, Fast-GNG, which applies the Add-If-Silent (AIS) idea to learn

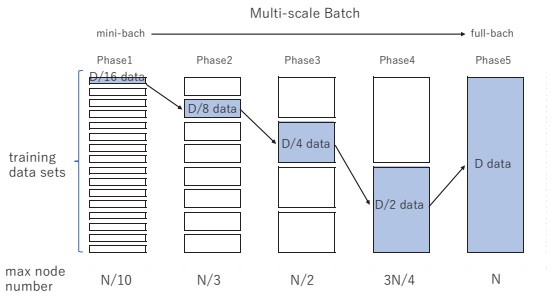


Fig. 1. Learning phase update. (*Multi-scale phase advances depending on the number of nodes generated, and the size of the referenced mini-batch increases as well*)

topological structures with fewer sampling times, has been proposed.

Fast-GNG is based on the algorithm proposed for MS-BL-GNG. Thus, the learning phase is set according to the number of nodes, and the mini-batch's size increases as the learning phase progresses. This enables learning that initially.
learns by capturing features of the data in broad strokes, and as learning progresses, captures detailed features by looking at the data as a whole (Fig. 1. shows the learning phase transitions and mini-batch sizes.). Fast-GNG also aims to obtain an accurate topological structure with fewer sampling times than MS-BL-GNG and achieves this by using the Add-If-Silent (AIS) idea.

The Fast-GNG algorithm is shown below, where $w_i$ is the reference vector corresponding to node i, $c_{ij}$ the edges between nodes i and j, and A is the entire set of nodes.

Step1. initialization

First, three initial nodes are randomly generated.

Step2. Minibatch initialization

$$\Delta w_i = 0, x_i = 0, c'_{ij} = 0 \tag{11}$$

$\Delta w_i$ is the temporal represents the number of updates in the batch, $c'_{ij}$ is the temporal edge in the batch, and $x_i$ is the number of times it was selected as the first or second winner node.

Step3. Mini-batch learning

Step3-1. Compute the first, second, and third winner nodes for the sample data v obtained from the input data.

$$S_1 = \arg\min_{i \in A} \|v - w_i\| \tag{12}$$

$$S_2 = \arg\min_{i \in A \setminus \{S_1\}} \|v - w_i\| \tag{13}$$

$$S_3 = \arg\min_{i \in A \setminus \{S_1, S_2\}} \|v - w_i\| \tag{14}$$

Step3-2. Update the error for the first and second winner nodes using the difference in distance between the sample data and the reference vector.

$$E_{s_1} \leftarrow E_{s_1} + \eta_1 \|v - w_{s_1}\| \tag{15}$$

$$E_{s_2} \leftarrow E_{s_2} + \eta_2 \|v - w_{s_2}\| \tag{16}$$

Where $\eta_1$ and $\eta_2$ are the learning coefficients and $E_i$ the error.

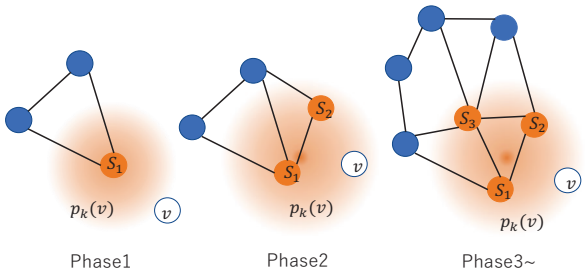


Fig. 2. Probability of node addition by AIS. (*Neighboring nodes are selected for input v, thereby changing the probability. The number of neighbor nodes used is increased by the multiscale phase*)

Step 3-3. create edges on the first winner node and the nodes that join it. Then update the number of selections for each node and update the temporal reference vector using the difference between the sample data and the reference vector.

$$x_{s_1} + +,\ x_j + + \left(j \in c_{s_1}\right), c'_{s_1,j} = 1 \tag{17}$$

$$\Delta w_{s_1} \leftarrow \Delta w_{s_1} + \eta_1 \left(v - w_{s_1}\right) \tag{18}$$

$$\Delta w_j \leftarrow \Delta w_j + \eta_2 \left(v - w_j\right) \quad if\ c_{s_1,j} = 1 \tag{19}$$

Step 3-4. Add nodes according to the AIS rule. Add a new node to the sampling data v according to the probability $p_k(v)$. The probability is calculated as follows (Fig. 2 shows What $p_k(v)$ represents.).

$$p_k(v) = \max\left(0, \tanh\left(z_k(v)/l_{Range}\right)\right) \tag{20}$$

$$z_k(v) = \sum_{i=1}^{k} \gamma_{k,i} d_{S_i} - l_k \tag{21}$$

Where $\gamma_{k,i}$ is a coefficient, and $l_k$ is as follows.

$$l_1 = l_{min},\ l_2 = l_{ave},\ l_3 = l_{ave} \tag{22}$$

$l_{min}$, $l_{ave}$, $l_{max}$ are the average, maximum, and minimum distance between connected nodes, respectively. And $l_{Range} = l_{min} - l_{max} + \varepsilon$. The sampling data v is added as a new node according to the probability calculated in this way. Then, a connection is made between the first winner node and the second winner node.

$$w_a = v \tag{23}$$

$$c'_{a,s_1} = 1, c'_{a,s_2} = 1 \tag{24}$$

Step4. Update mini-batch

Update the mini-batch when all data in the mini-batch have been calculated as follows. If not, it loops through Step 3. At this time, nodes not connected to other nodes are deleted, where β is the decay coefficient.

$$w_i \leftarrow w_i + \Delta w_i / x_i \quad if\ x_i > 0 \tag{25}$$

$$E_i \leftarrow E_i - \beta E_i (\forall i \in A) \tag{26}$$

$$c_{i,j} = \begin{cases} 1 & if\ c_{i,j} \geq 1 \\ 0 & otherwise \end{cases} \tag{27}$$

Step5. Delete nodes

The node with the least number of selections is deleted according to the times it was selected as the winner node in the mini-batch.

$$m_1 < \lambda_1 m_2 + \lambda_2 m_3 + \varepsilon \tag{28}$$

$$m_1 = \arg\min_{i \in A} x_i \tag{29}$$

$$m_2 = \arg\min_{i \in A \setminus \{m_1\}} x_i \tag{30}$$

$$m_3 = \arg\min_{i \in A \setminus \{m_1, m_2\}} x_i \tag{31}$$

where$\lambda_1$ and $\lambda_2$ are weights.

Step6. Add a node from the error.

The node with the largest error $q_1$and the node with the largest error having connectivity with that node $q_2$ are selected, and a new node is added between the two nodes to divide the two nodes by the ratio of their errors, and the reference vector, connectivity, and error are updated as follows.

$$q_1 = \arg\max_{i \in A} E_i \tag{32}$$

$$q_2 = \arg\max_{i \in c_{q_1}} E_i \tag{33}$$

$$w_r = \frac{1}{E_{all}} \left( E_{q_1} \cdot w_{q_1} + E_{q_2} \cdot w_{q_1} \right) \tag{34}$$

$$E_{all} = E_{q_1} + E_{q_2} \tag{35}$$

$$c_{r,q_1} = 1, c_{r,q_2} = 1, c_{q_1,q_2} = 0 \tag{36}$$

The errors of $q_1$ and $q_2$ are then reduced.

$$E_{q_1} \leftarrow E_{q_1} - \frac{E_{q_1}^2}{2E_{all}} \tag{37}$$

$$E_{q_2} \leftarrow E_2 - \frac{E_{q_2}^2}{2E_{all}} \tag{38}$$

$$E_r \leftarrow \frac{1}{2E_{all}} \left( E_{q_1}^2 + E_{q_2}^2 \right) \tag{39}$$

Step7. The phase level is updated when the number of nodes reaches the specified number for each phase. And repeat Steps 2~7 until the termination condition is met.

### *C. Proposed Method*

The proposed method uses SNN as described in A to remove noise to deal with asynchronous inputs, followed by learning with Fast-GNG as described in B to obtain the topological structure. Fig. 3 shows the flowchart of the entire system. This enables learning of the topological structure required to achieve ultra-real-time monitoring in response to the high noise level, which is a problem with event-based cameras, and asynchronous inputs, which are different from those of traditional cameras.

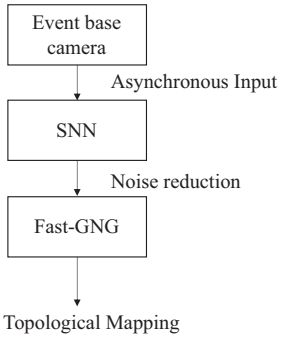


Fig. 3. Flowchart of the proposed method.

## III. EXPERIMENTS

Experiments will be conducted to confirm the effectiveness of the proposed method. The Event-based camera used in this experiment is a model PPS3MVCD (Fig. 4) manufactured by CenturyArks Co. The main specifications are summarized in Table I.

First, to confirm whether the SNN can remove noise, we will confirm the system's effectiveness by adding 10% and 20% white noise to the data set. The data set used in this experiment is a circular motion data set (Fig. 5), where SNN parameters are shown in Table II below, and the result of noise removal by SNN is shown in Fig. 6. The difference between the noiseless results and the SNN results was used to calculate the amount of noise present before and after the SNN process, which is summarized in Table III.

Fig. 4. Event-based camera used in this experiments

TABLE I. SPECIFICATIONS OF THE EVENT-BASED CAMERA

| Model | PPS3MVCD |
|---|---|
| Module effective pixels | 640×480 |
| Typical Latency | 200μs |
| Wide Dynamic Range | >120dB |
| Current consumption | 200~300mA |
| Power supply | 5.0V |

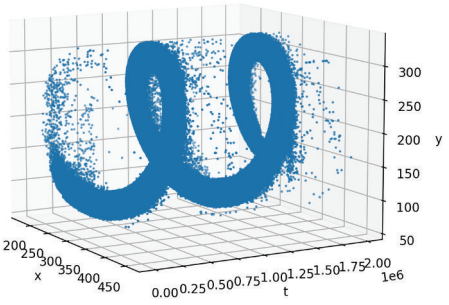


Fig. 5. Event-based camera dataset. (*where x and y are coordinates on the sensor and t is time[s]*)

Table III shows that from 9.07% to 0.17% for data with 10% noise and from 17.30% to 0.23% for data with 20%, indicating that the proposed SNN can remove most of the noise in the Event-based camera input and extract the event. The amount of noise in the raw data does not equal the amount of noise added here because of the coverage of the area where the event exists. On the other hand, Fig. 6 shows that the increasing amount of noise is concentrated in the area of the trajectory where the measurement target has been moving. This seems to have been caused by increased internal states of the SNN nodes in the trajectory area. There are also effects, the average internal status of each node in the SNN increasing with an increasing number of noises.

TABLE II. SNN PARAMETERS

| | |
|---|---|
| $w$ | 0.4 |
| $\theta$ | 30 |
| $R$ | 60 |
| $\gamma_1$ | 0.9995 |
| $\gamma_2$ | 0.9995 |
| $\gamma_3$ | 0.99999 |
| m | Eight neighbors |

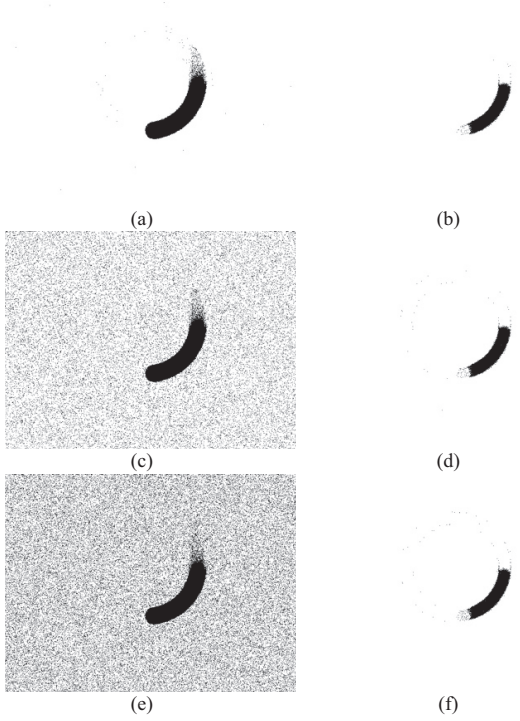


Fig. 6. Results of noise reduction by SNN. (*a: input data without noise, b: SNN output without noise,c: input data with 10% noise, d: SNN output with 10% noise, e: input data with 10% noise, f: SNN output with 10% noise*)

TABLE III. TABLE TYPE STYLES

| | Difference ratios [%] | |
|---|---|---|
| | Raw data | **SNN output** |
| Between noise, 10% and 0% | 9.07 | 0.17 |
| Between noise, 20% and 0% | 17.30 | 0.23 |

Although, it is difficult to assume that so much noise will be generated in real situations. And it seems to be possible to suppress it by appropriately controlling each parameter of the SNN. Although currently, each parameter needs to be set heuristically, this is an issue to be tackled in the future.

Next, to confirm the effectiveness of the proposed method as a whole, topological mapping using the proposed method is performed on a human face recorded in real life. The input data used were frontal videos of a person moving his head back and forth and videos of a person swinging his head to the left and right. The parameters of the Fast-GNG used are shown in Table IV (the parameters of the SNN are the same as in Table III). And the results of noise reduction by SNN and learning by GNG are shown in Fig.7, 8.

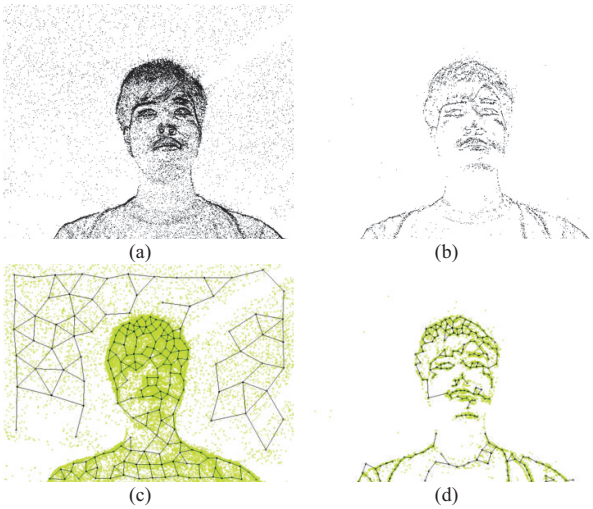


Fig. 7. Experimental results of face frontal view. (*a: Event-based camera data, b: Noise reduction using SNN, c: GNG learning without SNN, d: GNG learning with SNN*)

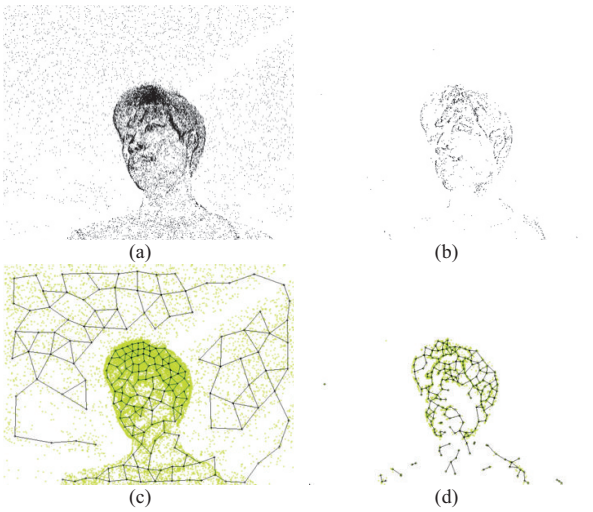


Fig. 8. Experimental results of swinging head. (*a: Event-based camera data, b: Noise reduction using SNN, c: GNG learning without SNN, d: GNG learning with SNN*)

TABLE IV. GNG PARAMETERS

| | |
|---|---|
| $\eta_1$ | 0.6 |
| $\eta_2$ | 0.4 |
| $\gamma_{1,1}$ | 1.0 |
| $\gamma_{2,1}$ | 1.0 |
| $\gamma_{2,2}$ | 0.5 |
| $\gamma_{3,1}$ | 1.0 |
| $\gamma_{3,2}$ | 0.66 |
| $\gamma_{3,3}$ | 0.33 |
| $\beta$ | 0.5 |
| $\lambda_1$ | 1 |
| $\lambda_2$ | 0.2 |
| $\varepsilon$ | 0.00001 |

Fig. 7(a), (b), Fig. 8(a), (b) shows that the noise reduction using SNN is working well, and only events can be extracted. And compared with the input image, it can be seen that the boundary area is also enhanced for the area where the event exists. These features are advantageous for feature extraction by learning GNGs, where both topological structure and probability distribution are affected. Comparing the topological structures obtained by the GNG, Fig. 7(c) and Fig. 8(c) show that in the case without noise reduction, nodes are generated for areas other than the area to be extracted, and only the rough shape of the face is captured, but the detailed structure is not learned. On the other hand, Fig. 7(d) Fig. 8(d) show that nodes are generated almost exclusively in the event area to be extracted through the SNN, and topological structures capture detailed features such as eyes, nose, and facial contours are obtained.

## IV. CONCLUSION

In this paper, we propose a methodology to realize an ultra-real-time monitoring system using an Event-based camera. SNN and fast-GNG are used to remove noise corresponding to asynchronous input/output, which is a characteristic of an Event-based camera. Then the topological mapping is used to extract features. We confirmed the effectiveness of the proposed methodology by conducting experiments. In future work, systems for recognition should be developed using the topological maps learned by the proposed method. The number of parameters that need to be determined for the SNN proposed in this paper has also increased, so a method to set such parameters is required appropriately. Also, as we have not considered the connection relationship between the nodes of the SNN, we believe that by setting and learning these parameters appropriately, it will be possible to construct a system that not only removes noise but also responds to specific inputs.

## REFERENCES


[1] N. Doteguchi, N. Kubota: "Topological Tracking for Mobility Support Robots Based on Multi-scale Batch-Learning Growing Neural Gas," 10th International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications, 2021.

[2] N. Kubota: "Multiscopic Topological Twin in Robotics," The 28th International Conference on Neural Information Processing, Bali, Indonesia, Online, December 8-12, 2021.

[3] N. Kubota, Y. Tomioka, M.Abe: "Temporalcoding in spiking neural network for gestures recognition of a partner robot," Proc. Joint 3rdInt. Conf. on Soft Computing and Intelligent Systems and 7th International Symposium on advanced Intelligent Systems, pp. 737-742, 2006.

[4] W.Maass and C.M.Bishop: "Pulsed Neural Networks," MIT Press, 1999.

[5] AW. Gerstner: "Spiking Neurons," In W. Maass and C. M. Bishop, editors, Pulsed Neural Networks, chapter 1, MIT Press, pp. 3-53, 1999.

[6] Bernd Fritzke: "A self-organizing network that can follow non-stationary distributions." Artificial Neural Networks—ICANN'97.Springer Berlin Heidelberg, 1997. 613-618.

[7] B. Fritzke: "A growing neural gas network learns topologies," Advances in Neural Information Processing Systems, Vol. 7, 1995, pp. 625-632.

[8] T . M. Martinetz and K. J. Schulten: "A "neural-gas" network learns topologies," Artificial Neural Networks, Vol. 1, 1991, pp. 397-402.

[9] B. Fritzke: "Unsupervised clustering with growing cell structures; Neural Networks," Vol. 2, 1991, pp 531–536.

[10] T. Kohonen: "Self-Organizing Maps," Springer, 2000.

[11] N. Mirehi, M. Tahmasbi, and A. T. Targhi, "Hand gesture recognition using topological features," Multimed Tools Appl, Vol.78, 2019, pp.13361-13386.

[12] P. M. Yanik et al.: "Use of Kinect depth data and Growing Neural Gas for gesture based robot control," 2012 6th International Conference on PervasiveComputing Technologies for Healthcare(PervasiveHealth) and Workshops, 2012, pp. 283-290.

[13] R. L. M. E. do Rego, A. F. R. Araujo and F. B. de Lima Neto: "Growing Self-Organizing Maps for Surface Reconstruction from unstructured Point Clouds," 2007 International Joint Conference on Neural Networks, 2007, pp. 1900-1905.

[14] A. Angelopoulou, J. Garcia-Rodriguez, S. Orts-Escolano, E. Kapetanios, X. Liang, B. Woll, A. Psarrou: "Evaluation of different chrominance models in the detection and reconstruction of faces and hands using the growing neural gas network," Pattern Analysis andApplications, Vol.22, No.4, 2019, pp.1667-1685.

[15] Y. Toda, W. Chin, N. Kubota: "Unsupervised neural network based topological learning from point clouds for map building," 2017International Symposium on Micro-NanoMechatronics and HumanScience (MHS), 2017.

[16] M. Iwasa, N. Kubota, andY. Toda: "Multi-scale Batch-Learning Growing Neural Gas for Topological Feature Extraction in Navigation of Mobility Support Robots," The 7th International Workshop on Advanced Computational Intelligence and Intelligent Informatics (IWACIII 2021), October 2021.