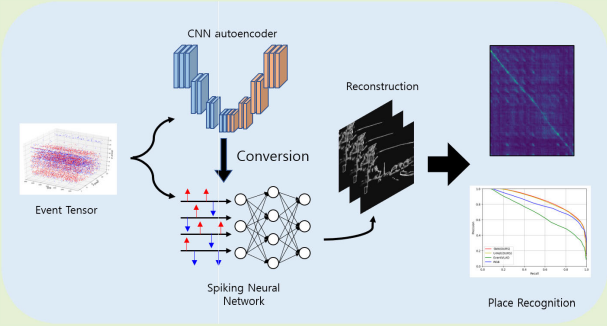
Ev-ReconNet: Visual Place Recognition Using Event Camera With Spiking Neural Networks

Hyeongi Lee, *Student Member, IEEE*, and Hyoseok Hwang, *Member, IEEE*

***Abstract*—In this article, we utilize the advantages of an event camera to tackle the visual place recognition (VPR) problem. The event camera's high measurement rate, low latency, and high dynamic range make it well-suited to overcome the limitations of conventional vision sensors. However, to apply the existing convolutional neural network (CNN)-based algorithms such as NetVLAD, the asynchronous event stream should be converted to a synchronous image frame, which causes a loss in temporal information. To address this problem, this article proposes a method that employs the asynchronous characteristic of spiking neural networks (SNNs) to leverage the temporal nature of event streams. The event stream is converted to event images and tensors in our preprocessing module. The SNN-based reconstruction networks, which are converted from CNNs, reconstruct edge images from event tensors regardless of external environment changes. VPR is conducted by matching features of the database and those from NetVLAD, which we used as a feature extraction network in this study. To evaluate the performance of VPR by comparing the previous methods for DDD17 and the Brisbane-Event-VPR dataset, experimental results demonstrate that the matching accuracy of the proposed method is better than previous methods, especially for datasets with adverse weather conditions. We also verify that the performance and energy efficiency are improved with SNNs over CNNs. Our code is available for download on https://github.com/AIRLABkhu/EvReconNet.**



***Index Terms*— Event camera, spiking neural networks (SNNs), visual place recognition (VPR).**

I. Introduction

EVENT camera is also called dynamic vision sensors and neuromorphic vision sensors that can capture high dynamic range (HDR) with low latency without motion blur [1]. Unlike conventional cameras, which synchronously upload all pixel values at a constant frame rate, event cameras record changes in pixel intensity asynchronously and each event's time, location, and polarity information. The HDR makes the event cameras much more robust against high-contrast illumination conditions [2]. They also have the advantage of lower power consumption compared to conventional cameras; therefore, event cameras are widely used in various fields, such as autonomous vehicles [3], robotics [4], and medical imaging [5].

Manuscript received 9 July 2023; revised 21 July 2023; accepted 23 July 2023. Date of publication 28 July 2023; date of current version 31 August 2023. This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korean Government [Ministry of Science and ICT (MSIT)] under Grant RS-2022-00167169. The associate editor coordinating the review of this article and approving it for publication was Dr. Avik Santra. *(Corresponding author: Hyoseok Hwang.)*

The authors are with the Department of Software Convergence, Kyung Hee University, Yongin 17104, Republic of Korea (e-mail: hyeongilee@khu.ac.kr; hyoseok@khu.ac.kr).

Digital Object Identifier 10.1109/JSEN.2023.3298828

Visual place recognition (VPR) is an essential task for the simultaneous localization and mapping (SLAM) of autonomous driving. It allows a vehicle to identify and navigate to previously visited locations by matching pairs between a query and a database [6]. Various approaches have been employed to solve VPR problems, including feature extraction and matching using handcrafted features [7], [8], [9], bag of words (BoW) [10], and deep learning-based methods [11]. Most existing methods for solving VPR problems were performed using images from conventional cameras, which are vulnerable to blur and rapid illumination changes. However, to improve the performance of VPR, the ability is necessary to match between images taken in the same place under various conditions regardless of weather, time, and amount of light (see Fig. 1). This explains why event cameras are better suited to solving VPR problems.

Several research studies have investigated applying event data to solve VPR problems [12], [13], [14], [15]. They mainly focused on reconstructing event data into the images to apply to the existing image-based VPR method such as NetVLAD [16]. Some approach [14] employs video reconstruction method [17] from the event stream to recognize places. A recent study has shown that reconstruction of only the edge image from which noise was removed had better

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

performance, rather than restoring the complete image from event stream [15]. We believe that there is room for further improvement as numerous studies suggest that spiking neural networks (SNNs) are more suitable for event-based steam. The main reason is that SNNs are designed explicitly for asynchronous processing, enabling them to handle each event as it arrives without requiring a fixed time interval [18]. Also, SNNs can use the event stream directly without losing temporal information or requiring multiple frames.

We propose a novel method for VPR from an event camera. Our approach consists of two successive models: the image reconstruction and the feature extraction networks. First, we reconstruct edge images from a stream camera using SNN-based reconstruction networks. To achieve this, a convolutional neural network (CNN)-based autoencoder that reconstructs an image optimized for place recognition was created and converted into an SNN. Then, reconstructed images are fed to feature extraction networks based on NetVLAD to extract features of the current location. The main contributions of this study are summarized as follows.

1) We propose a novel approach to process event data through SNNs and apply the advantages to VPR. To the best of our knowledge, this is the first study to apply an image reconstructed from an event stream using SNNs to VPR.
2) We experimentally demonstrate that our proposed VPR scheme, Ev-ReconNet-S, outperforms existing methods.
3) Our experiments demonstrate that the method based on SNNs, converted from CNNs, exhibits superior performance in VPR and, as per our analysis, is more energy-efficient.

II. Related Works

A. Event-Based Reconstruction

Since the advent of event cameras, research has continuously been conducted using event vision. Early reconstruction research is based on hand-engineered features. Bardow et al. [19] introduced an algorithm to simultaneously recover the motion field and reconstruct the intensity image, while the camera undergoes a generic motion through any scene. Munda et al. [20] created high-quality images by solving a certain mathematical model on a surface without having to estimate the optical flow. For the reconstruction of edge images, Lee et al. [21] applied a fitting plane algorithm to estimate the lifetime of the event using the intrapixel-area event considering the surface of active events (SAE). They activate an event until another event occurs in a nearby pixel of the pixel where the event occurred so that the shape of the edge is preserved. Mohamed et al. [22] calculated the lifetime of the event using the local plane fitting technique. They showed a result of reducing the response time to obtain edge images of the same sharpness compared to previous studies.

Deep neural networks (DNNs) have recently shown superior performance on an event-based image or video reconstruction. Wang et al. [23] utilized a generative adversarial network (GAN) to transform event streams into image brightness. Rebecq et al. [17] proposed E2VID, a high-performing video reconstruction method from event data. It was trained using a synthetic event dataset generated using ESIM [24] for a U-Net-based [25] network. Zhu et al. [18] proposed an SNN-based event video reconstruction method. They reconstructed video through the fact that spiking neurons have the potential to contain temporal information.

Fig. 1. Sample images from the DDD17 dataset taken from conventional cameras and event cameras. Even though they were taken in the same area and time of day, intensity images are sensitive to changes in illumination (first and second columns), and problems arise when the light is too bright or dark (third and fourth columns). However, the camera is more robust against these problems.

B. Visual Place Recognition

VPR is the problem of recognizing the same place despite significant changes in viewpoint and appearance. Early studies have employed a feature-based approach that can account for different environmental variations. For place recognition, Galvez-López and Tardos [10] proposed a BoW that builds code, such as clusters of features, and describes a scene by the code book. Milford and Wyeth [26] proposed SeqSLAM, which uses a local navigation sequence and matches images, removing the global matching process and alternatively matching with the nearing local images for efficiency and increased robustness at visual environment changes. As numerous studies have progressed, feature extraction methods using deep learning have begun to be introduced. The first work of applying CNNs to tackle the VPR problem is conducted by Chen et al. [27]. They utilized CNNs features extracted from a pretrained model called Overfeat [28]. Sünderhauf et al. [29] proposed a similar idea for image representation, vector of locally aggregated descriptors (VLAD) is a descriptor aggregation method for hand-engineered features. NetVLAD [16] mimics VLAD by using a CNN to obtain an image descriptor. A specifically designed pooling layer that implements the VLAD embedding and aggregation with differentiable operations, thus allowing end-to-end training of the network. Milford et al. [12] introduced a place recognition on event data using SeqSLAM. This study performed matching between event-based images and detecting loops at various velocity conditions indoors. Lee and Kim [15] implemented an image generation network suitable for VPR and used event data. They reconstructed edge images instead of intensity images from

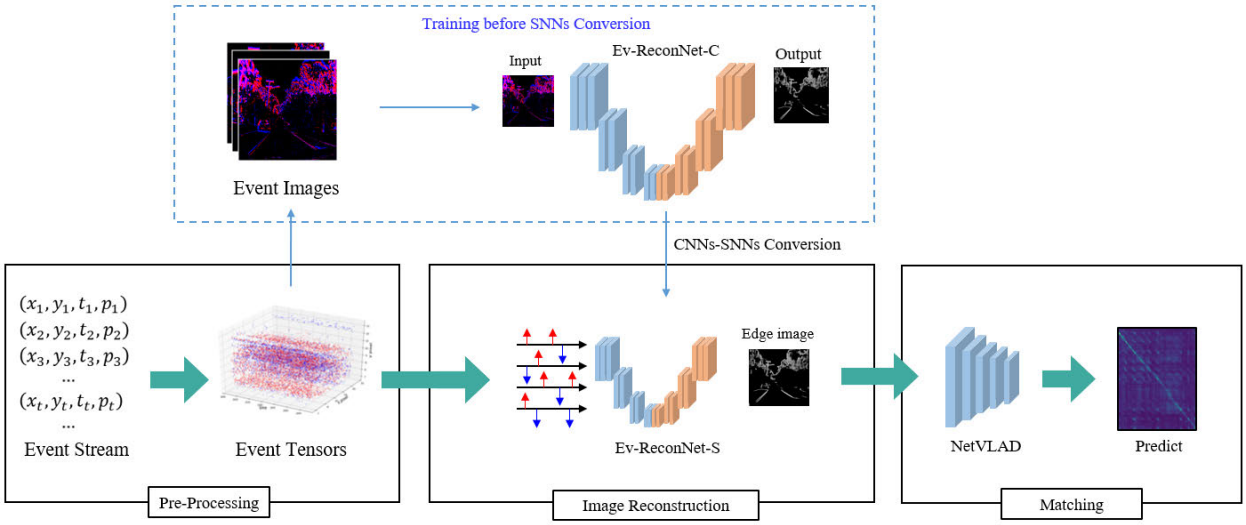


Fig. 2. Overview of the proposed method. Our method consists of three main parts: preprocessing, image reconstruction, and matching. For the implementation of the image reconstruction module, a CNN autoencoder based on U-Net was additionally used. The CNN-based autoencoder was converted to SNN using the Nengo toolkit and retrained using the Event Tensor. SNNs use a basic neuron model called LIF, and the overall structure is the same as the CNNs before conversion.

event data, which showed superior performance. However, the use of a CNN-based autoencoder for image reconstruction could potentially result in a loss of temporal information.

C. Spiking Neural Networks

SNNs are a type of artificial neural network (ANN) that more accurately emulate natural neural networks. The spiking neuron model incorporates an internal parameter known as the membrane voltage. When this membrane voltage reaches a certain threshold level, it generates an output spike, and the membrane voltage is reset to its resting potential. This output spike is then transmitted to another neuron, increasing the potential of the receiving neuron. SNNs are composed of a collection of these spiking neuron models. Due to this biomimetic structure, SNNs differ from traditional ANNs in receiving asynchronous spikes as input. This characteristic results in high compatibility with event cameras.

Spikeprop [30] is one of the first approaches to the supervised learning of SNNs, successfully applied to classification problems. Neural engineering framework (NEF) is one of the most utilized theoretical frameworks in neuromorphic computing [31]. As research advances, it has been recognized that, despite potential losses due to implementation, it is efficient to utilize SNNs converted from general ANNs such as CNNs due to their inherent advantages [32], [33], [34], [35], [36]. Rueckauer et al. [34] proposed a method to convert CNN operations, such as max pooling, softmax, batch normalization, and inception module for use in SNNs, and show the best results on datasets such as MNIST and CIFAR-10 in image classification. Stöckl and Maass [35] performed ANN–SNN transformation more efficiently with a novel mapping strategy using the few spikes neuron models (FS-neurons), which allows SNNs to temporarily exhibit complex activation functions with up to two spikes. Lopez-Randulfe et al. [36] applied time-coded neurons to existing ANNs and reduced the time complexity between synaptic connections. Nengo [37], a toolkit that approximates and transforms ANNs trained with TensorFlow into a spiking network, was also proposed by Applied Brain Research. Duwek et al. [38] proposed an approach to reconstruct image brightness from events based on the Laplacian and Poisson reconstruction using NEF-based SNNs through the Nengo framework.

III. Proposed Method

Our overall place recognition structure can be explained in three modules, as shown in Fig. 2. First, we briefly describe the data representation and preprocessing module. Second, we describe the detailed structure of the reconstruction networks and conversion SNNs from CNNs architecture. Finally, we will explain how the feature extraction networks are trained in a supervised method.

A. Event Representation and Preprocessing

Event cameras trigger events asynchronously by pixel. An event occurs when the following expression is satisfied:

$$|L(\mathbf{x}, t) - L(\mathbf{x}, t - \Delta t)| \geq C \tag{1}$$

where L is the log intensity, $\mathbf{x} = [x, y]^T$ is the pixel location, t stands for the timestamp, and C is the contrast threshold. The ith event can be represented as follows:

$$e_i = (\mathbf{x_i}, t_i, p_i) \tag{2}$$

where p is the polarity of the change of brightness. Since these individual event streams are difficult to apply directly to CNNs and SNNs, we converted the data into an image and a tensor format for CNNs and SNNs, respectively, in the preprocessing

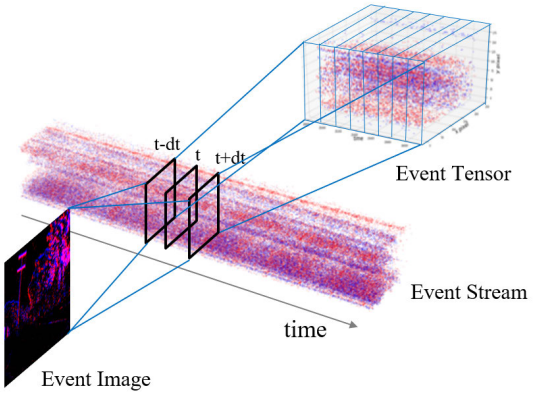


Fig. 3. Concept for converting event images and event tensors from the event stream.

step. Prior to conversion, noise and hot pixels were previously removed using dvs_tools.[1]

For a specific time t in the event stream, both event tensors and event images are converted by the sampling of the event data between $t - \Delta t$ and $t + \Delta t$, as shown in Fig. 3. To convert the event stream to event images, events between $t - \Delta t$ and $t + \Delta t$ are classified into two channels according to the polarity of each event. Each pixel's value is determined by the absolute difference between the count of positive and negative events that occurred in the corresponding pixel. Subsequently, the pixel value distribution is normalized to have a mean of zero and a standard deviation of one to finalize the image. Consequently, the event image is a single-channel matrix of size $H \times W$, which is utilized for training the CNNs.

The event tensor for SNNs is derived from the event stream in a similar manner, targeting all events that occur between $t - \Delta t$ and $t + \Delta t$, irrespective of polarity. The event data within the interval $t - \Delta t$ and $t + \Delta t$ are divided into T bins along the temporal axis, effectively subdividing the bin to have time intervals of $2\Delta t/T$. Events that belong to each bin are accumulated by one at a specific pixel position. This accumulation is then normalized by dividing by the maximum pixel value, ensuring that all pixel values range between 0 and 255. The resulting event tensor has dimensions of $T \times H \times W$. Based on our experimental results, we determined the values of T and Δt to be 10 and 0.1, respectively.

B. Image Reconstruction Networks

We propose Ev-ReconNet, which is an edge image reconstruction network for reconstructing edge images from event tensors. To achieve this, we first design networks based on CNNs and then convert them to SNNs. The CNN-based neural networks for image reconstruction are based on the autoencoder, which is composed of an encoder and a decoder such as U-Net [15], [17]. To reflect the differences in input data and training methods, we will refer to Ev-ReconvNet based on CNNs and SNNs as Ev-ReconvNet-C and Ev-ReconvNet-S, respectively.

The overall architecture of Ev-ReconvNet-C is shown in Fig. 4. The encoder networks of our reconstruction model are configured as follows. The input event image passes through convolution, max pooling, and dropout layers. The encoder block has a structure in which these three layers appear four times in succession. A 3×3 convolutional filter is used in the convolution layer, and the rectified linear unit (ReLU) function is used for all activation functions. The size of the max-pooling area is 2×2, and the dropout ratio is 0.1. We fixed the dimensions by reducing the padding size to 1. The decoder block consists of a convolution layer, a dropout layer, and an upsampling layer. Since the result of the encoder block is concatenated to the input, an upsampling layer consists of a transpose layer and a concatenate layer. The proposed architecture employs the skip connection, which is a characteristic of U-Net-based networks and shows performance improvements in image segmentation [15], [25]. Similar to the encoder block, the convolution filter has dimensions of 3×3, a padding size is 1, a dropout rate of 0.1 is employed, and the activation function used is the ReLU function. After the decoder block, it passes through two convolution layers to reconstruct one channel of the edge image.

We converted CNN-based reconstruction networks into SNN models. The conversion process from CNNs to SNNs and the subsequent training procedure were carried out using the NEF-based NengoDL library [37]. NEF is a widely recognized theoretical framework that is used in computational neuroscience and neuromorphic engineering to build large-scale functional neural simulations. Nengo, a Python-based neural compiler that translates high-level descriptions into low-level neural models, is built on the foundation of the NEF. During the conversion from CNNs to SNNs, we had to make several modifications to various functions. One of the significant changes involved the activation functions. The original ReLU activation functions were converted into spiking rectified linear activation functions. The output of these spiking activation functions is directly proportional to the quantity of the positive input spike, thereby creating a dynamic and responsive activation scheme. This spiking activation scheme is defined using two key parameters: a synaptic time constant and a maximal firing rate. In our model, we have chosen a synaptic time constant of 10 ms and a maximal firing rate of 100. These values were selected to ensure the optimal performance of the Ev-ReconNet-S model. Another significant modification was the choice of the loss function. Unlike CNNs, which typically use cross entropy as the loss function, SNNs utilize the mean squared error (mse). The reason for this change lies in the output of SNNs. The output of SNNs typically lies between 1 and 255, making the mse a more appropriate measure of the loss function. This adjustment further fine-tunes the performance of the Ev-ReconNet-S model, ensuring its accuracy and efficiency in the tasks it is designed to perform.

C. Feature Extraction Networks

In our VPR framework, we employ NetVLAD [16] for feature extraction networks. It is a structure in which a NetVLAD layer is added as a pooling layer after the CNN structure based on VGG16 [39]. The edge image from the reconstruction network was used as the input of NetVLAD. In order to utilize NetVLAD as a feature extraction module,

[1] https://github.com/cedric-scheerlinck/dvs_tools

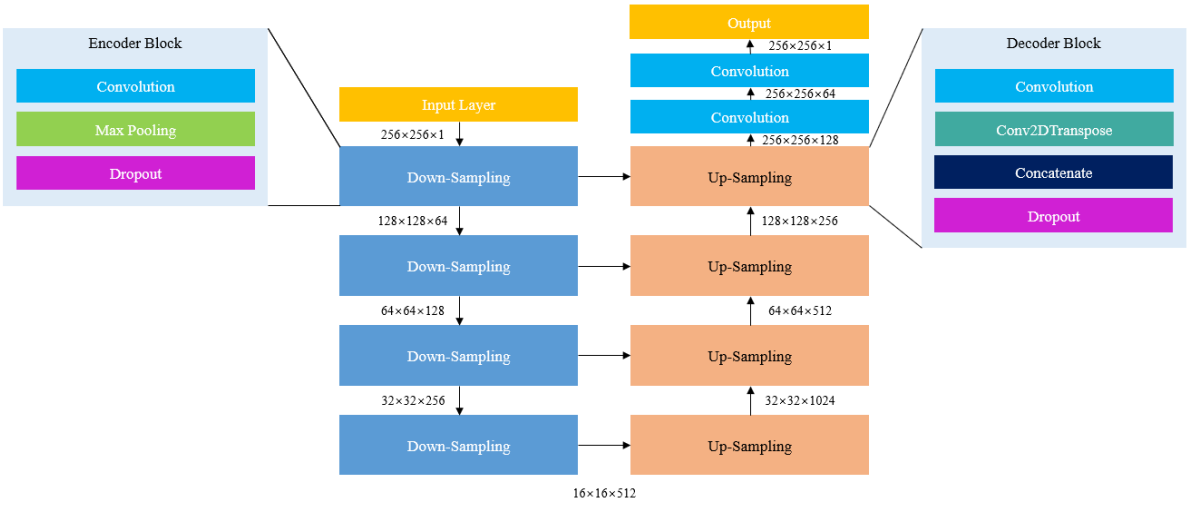


Fig. 4. Architecture of Ev-ReconNet-C. The Ev-ReconNet-S follows the same design except activation functions.

a clustering technique was needed before training to normalize the difference between each image. The number of NetVLAD clusters we used for our VPR framework is 64. The output of NetVLAD is a feature vector for the input data, which is a unit vector with 4096 dimensions. For the training of the feature extraction networks, the positive and the negative images of the reference image were set. A positive image is an image classified into the same category as the reference image, and a negative image is classified into a different category. As the loss function, we used triplet ranking loss using positive and negative images as

$$L(a, p, n) = \max(0, m + d(a, p) - d(a, n)) \tag{3}$$

where a is the feature vector of the reference image and p and n are the feature vectors of positive and negative images, respectively. d is the distance function. As the distance function, a pairwise distance with a norm degree of 2 was used. The margin m is set to 0.1.

D. Training VPR Framework

We will describe the learning methods by Ev-ReconNet-C and Ev-ReconNet-S, both of which are reconstruction networks. We first train Ev-ReconNet-C using event images before conversion to Ev-ReconNet-S. Then, Ev-ReconNet-S is trained with event tensors and pretrained parameters from Ev-ReconNet-C for a given number of epochs. To train reconstruction networks in a supervised manner, event streams and corresponding ground truth, i.e., edge images, are required. We solve this problem by using an event camera simulator [24] for converting the image dataset [40] into an event stream. The edge images are successfully obtained by applying the Canny edge extraction [41] method to intensity images. We convert the event image and event tensor from the same event stream so that both networks are trained on the same dataset. In both training processes, we set the learning rate to 0.001 and used the Adam optimizer. However, we conducted 100 and 30 epochs for the Ev-ReconNet-C and Ev-ReconNet-S, respectively.

In the case of feature extraction networks, training was conducted using database images. We chose positive pairs as images acquired from locations in geographical threshold distance, and negative pairs were selected randomly from sufficiently far locations. As the threshold distance, 70 m was used. We set the learning rate to 0.0001, the batch size was 4, and the stochastic gradient descent (SGD) optimizer was used. For the SGD optimizer, the weight decay value was 0.001, and the momentum was 0.9.

IV. Experimental Results

In this section, we outline the comprehensive performance of our proposed models and the results of our comparative analysis. First, we detail the localization methods and datasets employed in our experiments. Second, we describe the matching performance of the proposed approach by comparing previous methods, which use intensity images and edge images reconstructed by CNNs. Third, we analyze the differences between CNNs and their converted SNN counterparts by comparing their performance in image reconstruction, match accuracy, and energy consumption. For the experiments, we used graphics processing units (GPU)—Nvidia RTX 2080 Ti for the overall process. Also, an Intel[2] Core[3] i9-9900X CPU with eight processors was used.

A. Experiment Setup

1) Localization and Matching: We labeled each reconstructed image using GPS information where the original image was taken. First, the latitude and longitude data were transformed into $\boldsymbol{g}_k = [u, v]_k^T$, which follows the Universal Transverse Mercator (UTM) coordinate. u and v are the vertical and horizontal coordinates, respectively. We define the spatial distance d_{ij} between two reconstructed images $\mathcal{I}_i$ and $\mathcal{I}_j$ as the Euclidean distance of $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$. A true match is assumed if the spatial distance d_{ij} between the matched images is less than 70 m. To determine the distance v_{ij} of two output vectors $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$, the cosine similarity was used as

$$v_{ij} = \frac{\boldsymbol{f}_i \cdot \boldsymbol{f}_j}{\|\boldsymbol{f}_i\| \|\boldsymbol{f}_j\|}. \tag{4}$$

[2]Registered trademark.
[3]Trademarked.

We utilized recall@1 as the performance evaluation metric for matching accuracy. For each test image, we identified the single image exhibiting the highest similarity. If the location of this selected image falls within a 70 m radius of the true position, it is classified as a positive match. Consequently, the matching accuracy is determined by the ratio of the number of positive matches to the total number of images in the test dataset. Furthermore, to construct the precision–recall curve, we established a specific threshold value, denoted as τ. Any match with a v_{ij} value exceeding τ was classified as a positive match. We then adjusted the value of τ iteratively to generate the precision–recall curve.

2) Dataset: We used two groups of datasets for evaluation, one for training image reconstruction networks and the other for VPR evaluation. We trained methods such as E2VID and EventVLAD for image reconstruction from event data using the same datasets they used in their study. For example, E2VID and EventVLAD were trained using the MS COCO [42] and CARLA [43] datasets, respectively. The Oxford Robot Car dataset [40] is an open dataset that is taken with an RGB camera, which is frequently used in the autonomous driving field. This dataset is used to train the proposed methods, i.e., Ev-ReconNet-C and EV-ReconNet-S.

We used Pittsburgh [16], Brisbane-Event-VPR [14], and DDD17 [44] datasets to train NetVLAD. The Pittsburgh dataset comprises 58k images, which cover an area of 8.9 × 3.9 km. This dataset is used for training NetVLAD in their original study. Brisbane-Event-VPR and DDD17 datasets are used for evaluation validation and testing of VPR networks. This dataset contains six sequences of the same route in Brisbane. The route is approximately 8 km long and was traversed six times at different times of the day. We used five routes, except for the fourth route,[4] which was in poor condition due to very low illuminance among the six routes. We set the first route[5] as a database sequence, and the remainder were used as a query sequence. DDD-17dataset has over 12 h of a 346 × 260 pixel dynamic and active-pixel vision sensor (DAVIS) sensor recording highway and city driving in the daytime, evening, night, dry, and wet weather conditions, along with vehicle speed and GPS position. For VPR, overlapping paths were required, and four routes were used referring to [14].[6]

3) Comparing Methods: In our study, we evaluate the effectiveness of our proposed method by comparing it with various methods such as Raw (NetVLAD), E2VID, and EventVLAD. The Raw method denotes the NetVLAD method, where the intensity image of the dataset is used as input. We set the Raw method as a baseline for performance comparison. Other approaches, specifically E2VID and EventVLAD, employ event data as input, similar to our proposed method. Our goal with this method is to determine whether the use of event data provides a superior solution to the VPR problem than the RAW method. It is crucial that while both E2VID and EventVLAD utilize event data as input, their outputs are different. E2VID converts event data into an intensity image, while EventVLAD produces an edge image. By analyzing the experimental results of these methods, we aim to confirm the performance improvement provided by edge images in VPR tasks. Our suggested algorithm also processes event data and creates an edge image. However, it deviates from EventVLAD's fundamental architecture, as it is built on SNNs instead of CNNs. We aim to shed light on the performance variations between edge image creation using CNN and SNN methodologies through this difference. We briefly describe the comparison methods in Table I.

[4]20200427_181204-night.

[5]20200421_170039-sunset1.

[6]Rec1487350455 and rec1487417411 for the first set and rec148779465 and rec1487782014 for the second set.

TABLE I
DESCRIPTION OF COMPARING METHODS

	Raw	E2VID	EventVLAD	Ev-ReconNet-S (ours)
pretraining	-	MS COCO	CARLA	Oxford Robot Car
input data	grayscale	event data	event data	event data
transformed data (input to VPR)	-	grayscale	edge images	edge images
VPR method	NetVLAD	NetVLAD	NetVLAD	NetVLAD
architecture	CNNs	CNNs	CNNs, GRU	SNNs

TABLE II
MATCHING ACCURACY USING OFF-DATASET-BASED NETVLAD. IN THIS METHOD, NETVLAD IS TRAINED BY INTENSITY IMAGES OF PITTSBURGH DATASET ONLY

	Raw	**E2VID**	**EventVLAD**	**Ev-ReconNet-S (ours)**
DDD set1	20.95	54.39	64.49	**66.42**
DDD set2	87.38	96.29	96.39	**96.54**
sunset1-sunset2	75.98	78.38	80.92	**84.88**
sunset1-daytime	27.18	**31.37**	18.28	20.47
sunset1-morning	50.24	**50.75**	20.59	21.11
sunset1-sunrise	40.27	**48.83**	24.33	37.30

B. Match Performance

We conducted two experiments to evaluate matching performance with two types of feature extraction networks, i.e., off-dataset- and on-dataset-based NetVLAD. We used Brisbane-Event-VPR and DDD17 as test datasets in all experiments. The on-dataset-based method is to train and test the NetVLAD with the same dataset. At this time, the data used for training and testing were separated within the same dataset. The off-dataset method means that the Pittsburgh dataset is used for training, and the Brisbane-Event-VPR and DDD17 datasets are used for testing VPR networks. For both experimental setups, the image reconstruction methods were pretrained on datasets used in previous studies. Note that in the off-dataset method, the NetVLAD is trained using only intensity images, whereas the image reconstruction format of EventVLAD and the proposed method is the edge image. This is to verify that the edge images we generate are directly applicable to the VPR method trained on intensity images.

The experimental results using off-dataset NetVLAD are shown in Table II. The performance of E2VID methods, which reconstructs event stream to intensity image, is the best for the Brisbane-Event-VPR dataset. However, on the

TABLE III
MATCHING ACCURACY USING ON-DATASET-BASED NETVLAD

	Raw	E2VID	EventVLAD	Ev-ReconNet-S (ours)
DDD set1	35.16	66.08	72.24	**72.47**
DDD set2	96.36	97.78	96.92	**98.46**
sunset1-sunset2	87.84	89.57	84.93	**95.73**
sunset1-daytime	36.37	**36.39**	26.86	36.37
sunset1-morning	**61.39**	60.81	32.25	44.60
sunset1-sunrise	64.97	62.20	47.59	**66.08**

DDD17 sets and sunset1-sunset2 data, accuracy when using Ev-ReconNet-S is better than other methods, despite the edge images we recovered from the event tensor having a different format than the intensity images from which we trained the VPR networks. In particular, the performance improvement was noticeable in the DDD set1, presumably due to the significant change in the environment between the database and the query. As can be seen in the third column of Fig. 1, the intensity of sunlight was so strong that there were frames that looked like blank images with conventional cameras, but event cameras could recognize them without any problems due to the HDR in the DDD set1. Although the DDD set2 and sunset1-sunset2 datasets were not significantly affected by the light intensity, the accuracy was relatively high because the event camera effectively contained the edge information. The sunset1-morning dataset exhibited a different direction of sunlight compared to the sunset1 data. However, the brightness of the light was largely similar in both datasets, resulting in minimal differences in the intensity of images between the two. Since E2VID was also trained based on RGB images, similar results were obtained when using intensity images.

The matching performance when using the on-dataset-based NetVLAD is described in Table III. The performance of the proposed method was the best on all datasets except the sunset1-morning and sunset1-daytime datasets. However, the sunset1-daytime dataset differed only by 0.2%P compared to the best-performed method (E2VID). When utilizing NetVLAD on the DDD set1, it encountered issues with dynamic range when processing intensity images. Conversely, when event-based data were utilized, the issue was resolved, as seen in previous experiments, and the reconstructed images produced by our approach demonstrated the best performance. To clarify, training with event data can readily overcome current performance limitations, and issues related to timing can be resolved by offline learning using NetVLAD. However, in the sunset1-morning dataset, the method using only intensity data still shows the best performance, which means that there are defects in the event data and still problems to be solved. Except for some cases, we found that the performance was better in the order of Ev-ReconNet (SNNs), E2VID, EventVLAD, and NetVLAD. We can see that edge-based images have better match performance than intensity images for the same dataset, as shown in Fig. 5.

Figs. 6 and 7 show a precision–recall curve when the value of the threshold τ changes for true prediction. Looking at the precision–recall curve of DDD set1, the area under the curve (AUC) of intensity images is significantly less than ours. DDD set2 also shows that our algorithm makes a slightly

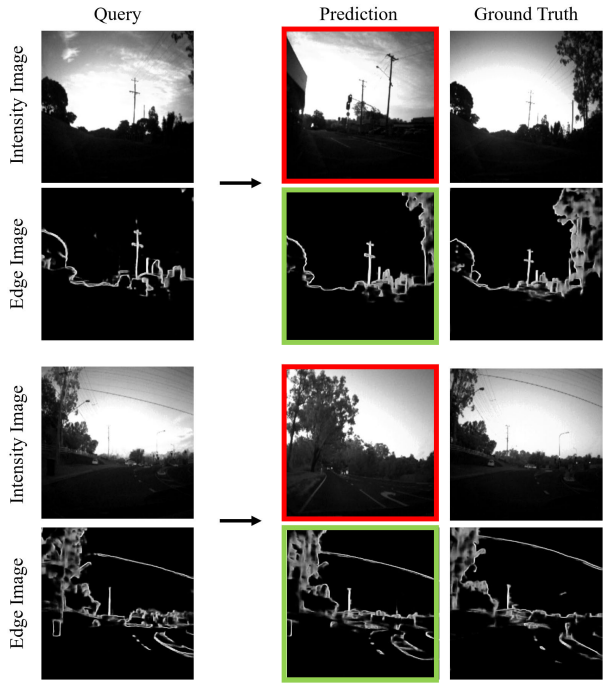


Fig. 5. Matching results using reconstructed edge events from the Brisbane-Event-VPR dataset. The matching of the intensity image failed (red), but the reconstructed image using the event data succeeded in matching (green). In both cases, the on-dataset-based network was used for matching. The intensity image of the upper row was input to NetVLAD, and the intensity image of the lower row was reconstructed using Ev-ReconNet and then input to NetVLAD.

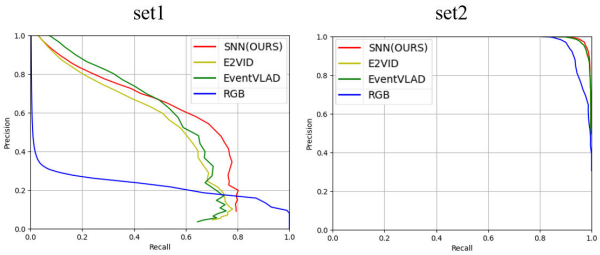


Fig. 6. Precision–recall curves on the DDD-17 dataset using on-dataset-based NetVLAD. Our proposed method, the Ev-ReconNet-S shows better performance than the rest of the algorithms.

larger AUC. For the Brisbane dataset, our algorithm increased the performance than others for the sunset1-sunset2 and sunset1-sunrise datasets. With the sunset1-daytime dataset, our method showed better precision, but when the recall increased (when the τ value decreased), it showed poor performance compared to the NetVLAD.

C. Performance Comparison Between CNNs and SNNs

We conducted additional experiments to compare the performance of methods of EvReconNet based on CNNs and SNNs. These experiments were designed to evaluate the performance of image reconstruction and VPR accuracy using both methods under identical conditions. Finally, we conducted an analysis to compare the energy consumption of the two methods.

1) Image Reconstruction Performance: We conducted an additional experiment to validate the performance of our reconstruction method. In this experiment, we use a Canny

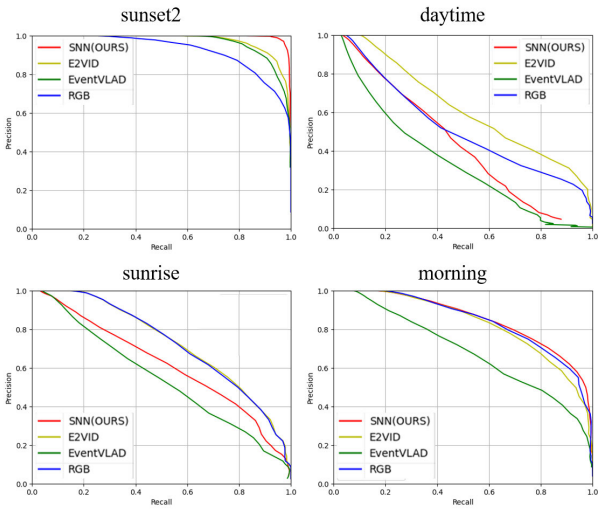


Fig. 7. Precision–recall curves on the Brisbane-Event-VPR dataset using on-dataset-based NetVLAD. Except sunset1-morning as query data, our algorithm shows similar or superior performance to the other methods.

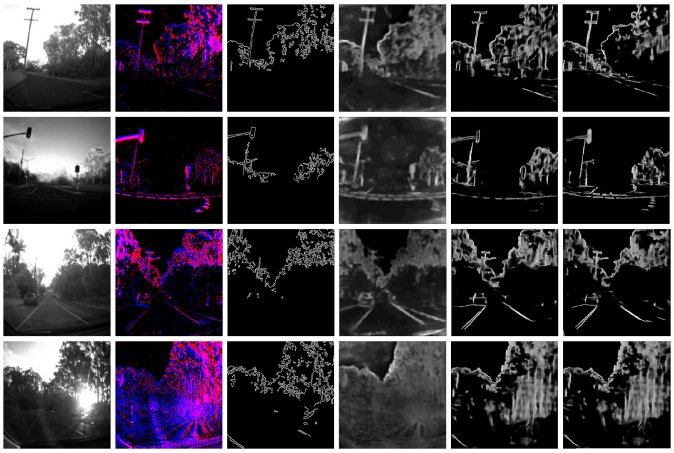


Fig. 8. Samples of the reconstructed edges from the Brisbane-Event-VPR dataset. Raw intensity images (the first column), event images that represent their polarities (the second column), Canny edge images (the third column), and reconstructed edge images using EventVLAD, Ev-ReconNet-C, and Ev-ReconNet-S.

edge detector to obtain edge images for ground truth. The reconstructed images of comparing methods are shown in Fig. 8. We observe that even in similar locations, there are large differences between intensity images depending on the time of day, but less so for edge images. We measured the multiscale structural similarity index (MS-SSIM) [15], [45] between the ground truth and reconstructed edge image to evaluate the reconstruction performance of the proposed method, of which the formula is given as follows:

$$L^{\text{MS-SSIM}} = 1 - \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C1} \prod^{M} \frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{5}$$

where C_1 and C_2 are constants, μ and σ are the mean and standard deviation of each image, and M is the scale pyramid. Table IV shows the reconstruction performance of Ev-ReconvNet-C and the Ev-ReconvNet-S. The results demonstrate that the reconstructed image of Ev-ReconvNet-S is closer to the ground truth in all datasets. We analyze the reason for these results that the event tensor has negligible loss of temporal information compared to the event image.

TABLE IV
SSIM ALONG OUR RECONSTRUCTION AND EVENTVLAD FOR EACH DATASET

Dataset	sunset1	sunset2	daytime	morning	sunrise
Ev-ReconNet-C	0.797	0.757	0.722	0.799	0.798
Ev-ReconNet-S	**0.864**	**0.830**	**0.812**	**0.855**	**0.857**

TABLE V
MATCHING ACCURACY COMPARISON OF EV-RECONNET-C AND EV-RECONNET-S

	off-dataset-training		on-dataset-training	
	Ev-ReconNet (CNNs)	Ev-ReconNet (SNNs)	Ev-ReconNet (CNNs)	Ev-ReconNet (SNNs)
DDD set1	64.38	**66.42**	**72.47**	**72.47**
DDD set2	**96.55**	96.54	96.55	**98.46**
sunset1-sunset2	82.41	**84.88**	94.65	**95.73**
sunset1-daytime	**23.05**	20.47	36.06	**36.37**
sunset1-morning	20.81	**21.11**	42.57	**44.60**
sunset1-sunrise	33.21	**37.30**	64.75	**66.08**

2) Matching Accuracy Comparison: We conducted additional experiments to validate the performance gap between CNNs and SNNs from the same architecture. The experimental results are shown in Table V. We confirmed that conversion to SNNs has higher performance, except for DDD set2 and senset1-daytime datasets when using pretrained parameters. In particular, SNNs trained using a database showed the same or better results than CNNs in all datasets. In addition, as the reconstruction of the edge image was well performed, it was confirmed that the VPR performance also increased. We analyze the main reason in two aspects. The first reason is the structural difference between Ev-ReconNet-C and Ev-ReconNet-S. In general, the architecture of the SNNs is identical to that of CNN; however, there are a few differences. SNNs use a basic neuron model called leaky integrate and fire (LIF). The converted SNNs employ mse as the loss function, whereas the original CNNs use the cross-entropy loss. This implies that the performance may vary when CNNs are converted to SNNs with an identical structure.

3) Energy Consumption: We conducted a comparative analysis of the energy consumption between the proposed methods based on CNNs and SNNs. For this purpose, we employed the Keras-Spiking framework to simulate the energy estimation of the CNN on the Intel-I7-4960X CPU and the Nvidia GTX Titan Black GPU, as well as the SNN on the Intel Loihi CPU (a neuromorphic chip). Our analysis is based on several assumptions by following the same conditions as in [46]. Table VI outlines the energy consumption for each method. The analysis results indicate that, when executed on a neuromorphic chip, the proposed method based on SNNs exhibits the least energy consumption. This can be one of the crucial advantages of the proposed method.

TABLE VI
COMPARE ENERGY CONSUMPTION ANALYTICS. THE MEASUREMENT UNIT IS TOTAL ENERGY PER INFERENCE (JOULES/INFERENCE)

	E2VID	**EventVLAD**	**Ev-ReconNet-C**	**Ev-ReconNet-S**
Energy consumption (J/inf)	3.28	6.13	4.74	4.73×10^{-3}

TABLE VII
ACCURACY USING VARIOUS T AND Δt VALUES. T AND Δt VALUES ARE EVALUATED WHILE KEEPING Δt AND T CONSTANT AT 0.1 AND 10, RESPECTIVELY

	T				Δt			
	5	10	15	20	0.05	0.10	0.15	0.20
DDD Set1	70.89	**72.47**	72.43	72.43	72.23	**72.47**	72.38	72.10
DDD Set2	86.84	**98.46**	98.33	98.41	98.24	**98.46**	98.27	98.23
sunset1-sunset2	95.02	**95.73**	95.56	95.60	95.17	**95.73**	95.70	95.63
sunset1-daytime	36.30	**36.37**	36.35	36.35	35.50	**36.37**	36.39	35.82
sunset1-morning	42.44	**44.60**	44.80	44.80	44.58	**44.60**	44.42	44.33
sunset1-sunrise	65.56	**66.08**	66.02	66.02	65.22	**66.08**	65.83	65.77

D. Accuracy Affected Parameters of Event Tensor

We conducted extensive experiments to analyze the effect of event tensors' parameters on our method's performance. We experimentally tuned the number of bins T and the marginal time Δt when converting the event tensor from the event stream at time t. In the first experiment, we tested various T values 5, 10, 15, and 20 while keeping Δt constant at 0.1. In the second experiment, we varied Δt from 0.05 to 0.20 while maintaining T at a constant value of 10. The resulting dataset accuracies using T and Δt are shown in Table VII. Based on these experiments, we concluded that T and Δt are crucial performance-affecting parameters.

V. CONCLUSION

In this study, we investigated the use of event camera data for VPR tasks and proposed an SNN approach that addresses the issue of temporal information loss during the discretization process of converting event data into images. Specifically, our approach leverages the spatiotemporal processing capabilities of SNNs to directly process the raw event data without the need for image conversion. Our experimental results demonstrate that the SNN approach can achieve competitive performance on the benchmark dataset and outperforms a method based on event cameras combined with convolutional neural networks. Importantly, our approach preserves the temporal information of the event data, which we believe contributes to its superior performance compared to methods that rely on image conversion. Our future work will focus on speeding up and further improving the performance of the converted SNN model.

REFERENCES

[1] R. Berner, C. Brandli, M. Yang, S.-C. Liu, and T. Delbruck, "A 240×180 10 mW 12us latency sparse-output vision sensor for mobile applications," in *Proc. Symp. VLSI Circuits*, Jun. 2013, pp. C186–C187.

[2] L. Wang, T.-K. Kim, and K.-J. Yoon, "EventSR: From asynchronous events to image reconstruction, restoration, and super-resolution via end-to-end adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8312–8322.

[3] A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5419–5427.

[4] G. Gallego et al., "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, Jan. 2022.

[5] A. Zihao Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," 2018, *arXiv:1802.06898*.

[6] S. Lowry et al., "Visual place recognition: A survey," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 1–19, Feb. 2016.

[7] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, Sep. 1999, pp. 1150–1157.

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Proc. 9th Eur. Conf. Comput. Vis.*, vol. 3951, May 2006, pp. 404–417.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.

[10] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012.

[11] X. Zhang, L. Wang, and Y. Su, "Visual place recognition: A survey from deep learning perspective," *Pattern Recognit.*, vol. 113, May 2021, Art. no. 107760.

[12] M. Milford et al., "Place recognition with event-based cameras and a neural implementation of SeqSLAM," 2015, *arXiv:1505.04548*.

[13] D. Kong, Z. Fang, H. Li, K. Hou, S. Coleman, and D. Kerr, "Event-VPR: End-to-end weakly supervised network architecture for event-based visual place recognition," 2020, *arXiv:2011.03290*.

[14] T. Fischer and M. Milford, "Event-based visual place recognition with ensembles of temporal windows," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6924–6931, Oct. 2020.

[15] A. J. Lee and A. Kim, "EventVLAD: Visual place recognition with reconstructed edges from event cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 2247–2252.

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

[17] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "High speed and high dynamic range video with an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 6, pp. 1964–1980, Jun. 2021.

[18] L. Zhu, X. Wang, Y. Chang, J. Li, T. Huang, and Y. Tian, "Event-based video reconstruction via potential-assisted spiking neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 3584–3594.

[19] P. Bardow, A. J. Davison, and S. Leutenegger, "Simultaneous optical flow and intensity estimation from an event camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 884–892.

[20] G. Munda, C. Reinbacher, and T. Pock, "Real-time intensity-image reconstruction for event cameras using manifold regularisation," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1381–1393, Dec. 2018.

[21] S. Lee, H. Kim, and H. J. Kim, "Edge detection for event cameras using Intra-pixel-area events," 2019, *arXiv:1907.07469*.

[22] S. A. Mohamed, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, "Towards real-time edge detection for event cameras based on lifetime and dynamic slicing," in *Proc. Int. Conf. Artif. Intell. Comput. Vis. (AICV)*. Cham, Switzerland: Springer, 2020, pp. 584–593.

[23] L. Wang, I. S. M. Mostafavi, Y.-S. Ho, and K.-J. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10073–10082.

[24] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: An open event camera simulator," in *Proc. Conf. Robot Learn.*, 2018, pp. 969–982.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Munich, Germany: Springer, Oct. 2015, pp. 234–241.

[26] M. J. Milford and Gordon. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 1643–1649.

[27] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional neural network-based place recognition," 2014, *arXiv:1411.1509*.

[28] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*.

[29] N. Sunderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of ConvNet features for place recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 4297–4304.

[30] S. M. Bohte, J. N. Kok, and J. A. L. Poutré, "SpikeProp: Backpropagation for networks of spiking neurons," in *Proc. ESANN*, vol. 48. Bruges, 2000, pp. 419–424.

[31] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, Dec. 1997.

[32] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Netw.*, vol. 111, pp. 47–63, Mar. 2019.

[33] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. M. McGinnity, "A review of learning in biologically plausible spiking neural networks," *Neural Netw.*, vol. 122, pp. 253–272, Feb. 2020.

[34] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers Neurosci.*, vol. 11, p. 682, Dec. 2017.

[35] C. Stöckl and W. Maass, "Optimized spiking neurons can classify images with high accuracy through temporal coding with two spikes," *Nature Mach. Intell.*, vol. 3, no. 3, pp. 230–238, Mar. 2021.

[36] J. López-Randulfe, N. Reeb, and A. Knoll, "Conversion of ConvNets to spiking neural networks with less than one spike per neuron," in *Proc. Conf. Cognit. Comput. Neurosci.*, 2022, pp. 553–555.

[37] T. Bekolay et al., "Nengo: A Python tool for building large-scale functional brain models," *Frontiers Neuroinform.*, vol. 7, p. 48, Jan. 2014.

[38] H. C. Duwek, A. Shalumov, and E. E. Tsur, "Image reconstruction from neuromorphic event cameras using Laplacian-prediction and Poisson integration with spiking and artificial neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 1333–1341.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[40] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 3–15, Jan. 2017.

[41] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[42] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Zurich, Switzerland: Springer, Sep. 2014, pp. 740–755.

[43] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Conf. robot Learn.*, 2017, pp. 1–16.

[44] J. Binas, D. Neil, S.-C. Liu, and T. Delbruck, "DDD17: End-to-end Davis driving dataset," 2017, *arXiv:1711.01458*.

[45] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017.

[46] T. Q. Nguyen, Q. T. Pham, P. C. Hoang, Q. H. Dang, D. M. Nguyen, and H. H. Nguyen, "An improved spiking network conversion for image classification," in *Proc. Int. Conf. Multimedia Anal. Pattern Recognit. (MAPR)*, Oct. 2021, pp. 1–6.

Hyeongi Lee (Student Member, IEEE) received the B.E. degree from the Department of Software Convergence and the Department of Mechanical Engineering, Kyung Hee University, Yongin, South Korea, in 2023.

In 2023, he joined Hyundai Mobis, Inc., Seoul, South Korea, where he is engaged in research and development in the field of vehicle communication systems. He is currently a Researcher with the Product Development Ecosystem Cell, Hyundai Mobis, Inc.

Hyoseok Hwang (Member, IEEE) received the B.S. degree in mechanical engineering from Yonsei University, Seoul, South Korea, in 2004, the M.S. degree in robotics, and the Ph.D. degree from the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2009 and 2017, respectively.

From 2009 to 2018, he worked at the Samsung Advanced Institute of Technology (SAIT), Samsung Electronics, Suwon, South Korea, as a Senior Researcher. He is currently an Assistant Professor with the Department of Software Convergence, Kyung Hee University, Yongin, South Korea. His research interests focused on computer vision and machine learning, which spans over 3-D perception and reconstruction for intelligent robot and autonomous systems.