Spiking Neural Networks for Active Time-Resolved SPAD Imaging

Yang Lin Edoardo Charbon
Ecole polytechnique fédérale de Lausanne
Neuchâtel, Switzerland
{yang.lin, edoardo.charbon}@epfl.ch

Abstract

Single-photon avalanche diodes (SPADs) are detectors capable of capturing single photons and of performing photon counting. SPADs have an exceptional temporal resolution and are thus highly suitable for time-resolved imaging applications. Applications span from biomedical research to consumers with SPADs integrated in smartphones and mixed-reality headsets. While conventional SPAD imaging systems typically employ photon time-tagging and histogram-building in the workflow, the pulse signal output of a SPAD naturally lends itself as input to spiking neural networks (SNNs). Leveraging this potential, SNNs offer real-time, energy-efficient, and intelligent processing with high throughput. In this paper, we propose two SNN frameworks, namely the Transporter SNN and the Reversed Start-stop SNN, along with corresponding hardware schemes for active time-resolved SPAD imaging. These frameworks convert phase-coded spike trains into density- and interspike-interval-coded ones, enabling training with rate-based warm-up and Surrogate Gradient. The SNNs are evaluated on fluorescence lifetime imaging. The results demonstrate that the accuracy of shallow SNNs is on par with established benchmarks. Our vision is to integrate SNNs in SPAD sensors and to explore advanced SNNs within the proposed schemes for high-level applications.

1. Introduction

Single-photon avalanche diodes (SPADs) are solid-state photodetectors capable of detecting single photons [7, 37]. Low timing jitter has made SPAD a popular technology to measure arrival time of the incident photons with high precision [26]. Due to successful implementation in CMOS technology, SPADs are highly reproducible and can be manufactured reliably with high levels of miniaturization [28]. CMOS SPADs may be arranged into large arrays for wide-field imaging and, thanks to their digital nature, processing may be added on chip near each pixel. Compared to other single-photon detectors such as photomultiplier tubes (PMTs) and electron-multiplying charge-coupled devices (EMCCDs), the low cost and high timing resolution of SPADs make them suitable for time-resolved biophotonics applications such as fluorescence lifetime imaging microscopy (FLIM) and positron emission tomography (PET), but also in machine vision applications, such as light ranging and detection (LiDAR) [31, 33]. In recent years, SPAD sensors gradually entered the consumer market. In 2015, STMicroelectronics introduced SPAD-based proximity sensors that are now in most smartphones. In 2020, Apple Inc. first introduced SPAD image sensors into consumer-grade devices (iPhone Pro and iPad Pro) as part of a LiDAR system [1], and further extended to the latest mixed reality headset [2]. This year, Sony also announced IMX611, an affordable SPAD sensor for smartphones [32]. Canon Inc. announced that it was developing the first interchangeable-lens SPAD-based camera [8]. These emerging applications usher in opportunities and challenges of developing computer vision systems and algorithms for SPAD image sensors.

In a SPAD, upon the arrival of a photon, the avalanche breakdown is triggered within the SPAD, resulting in a pulse at the output voltage. The pulses are counted in passive imaging to obtain intensity information or they are time-tagged in active imaging to obtain timing information. Figure 1 demonstrates the conventional workflow of active time-resolved SPAD imaging. Substantial amounts of data are generated in this processing, posing challenges to data transfer, processing, and storage on both hardware and software levels. While artificial neural networks (ANNs) have been utilized to process data generated by SPADs, one further step could be made to integrate spiking neural networks (SNNs) into vision systems, considering that a SNN takes the spike as input. An SNN can be thus directly connected to the SPAD, eliminating time-to-digital converters (TDCs) and histograms as well as enabling end-to-end learning and inference. Implemented on the SPAD sensor, the SNN-based processing can reduce latency, bandwidth, and power consumption, while realizing real-time and intelligent analysis.

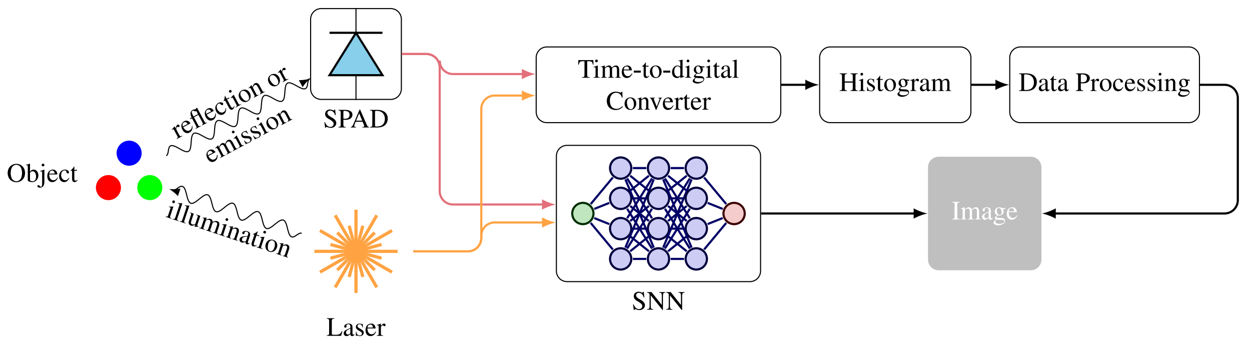


Figure 1. Workflow of the traditional SPAD TCSPC system and the SNN-coupled SPAD system. In a traditional SPAD TCSPC system, the laser illuminates the object and sends a reference signal to the time-to-digital converters (TDCs). The object reflects or emits light, which is detected by the SPAD. Reference pulse signals from the laser and detection pulse signals from the SPAD are directed to time-to-digital converters (TDCs) for time-tagging. The timestamps are often histogrammed and transmitted to a PC for data processing, where the image is reconstructed.

Though, in principle, SNNs can be used to process signals from both active and passive SPAD imaging, they are essentially different tasks. In an active imaging setup, the target is repeatedly illuminated by a laser or photodiode. The information is represented by the difference between the arrival time of the photon and the reference, which is basically phase-coded. In a passive imaging setup, spiking generated by incoming photons is a Poisson process, which is rate-coded. Therefore, from the signal processing perspective, the SPAD sensor is no more than a traditional CMOS sensor except for extreme photosensitivity and high frame rate. The existing algorithms that adopt a Poisson encoder are supposed to work for passive SPAD imaging as well. In this work, we only focus on active imaging with SPAD sensors.

ANNs take inspiration from neurons and synapses of biological neural networks and achieve stunning success in the last decade. By introducing the third factor, i.e. temporal dynamics, the SNN is seen to mimic biological neural networks and a promising model to supersede ANNs. It has been successfully implemented in some vision systems [11, 19, 27, 35]. To construct an SNN, several aspects must be considered, including the encoder, spiking neuron model, network topology, decoder, and training methods. Aiming at in-sensor and near-sensor implementation, the former four aspects cannot be chosen arbitrarily due to hardware limitations. Dedicated hardware is needed for specific coding schemes and spiking neurons. Training SNN is a challenging problem due to the non-differentiability of the firing function [10, 24]. Most existing methods set constraints on the coding scheme or the firing behavior. ANN-to-SNN conversion assumes rate-based coding while backpropagation through spike time assumes latency coding and often limits the number of firings [4, 5, 9, 23, 29]. Recently, surrogate gradients have emerged as a promising training method that is applicable across various coding schemes [14, 20, 22, 25].

Despite the difficulty of efficient training, SNNs have been used in several applications. ANNs are transformed into SNNs to reduce energy consumption and running time [13, 21]. SNNs have been used for detectors such as electroencephalogram (EEG) [34] and event cameras [12,18,27] for various applications. The community, however, focuses more on classification than regression [11,27], more on still images than sequences, and more on rate-based coding than other coding schemes, while SPAD-based applications are more of phase coding, sequential processing, and regression. Only a few studies have been performed on the SNNs for SPAD and other single-photon detectors. In [35] it was first proposed to use SNNs to process single-photon signals; the authors use SNN to process LiDAR raw data for object detection. However, it is assumed that all the spikes come within one repetition period, which is not applicable to other tasks such as FLIM. [30] implemented SNNs with SPAD sensors, but only works on handwritten digit recognition with simulation data as passive imaging. To this date, there still lacks a general framework for constructing SNNs for active SPAD imaging.

In this paper, we present two general SNN frameworks, Transporter SNN and Reversed Start-stop (RS) SNN, designed specifically for active time-resolved SPAD imaging. With ring oscillator-based hardware, Transporter SNN folds the time domain across multiple repetition periods. Phase-coded spike trains are converted into density-coded spike trains, reducing the sparsity and length of the spike trains and thus facilitating training and inference processes. With flip-flop-based hardware, RS SNN converts phase-coded spike trains into interspike-interval (ISI) -coded ones, combining the advantages of phase and rate coding and enabling more efficient training and inference. Surrogate gradients and rate-based warm-up are utilized to train the SNN. These frameworks are evaluated through experimentation on a classic application of SPAD image sensors, namely fluorescence lifetime estimation.

2. Problem Formulation

Active SPAD imagers naturally produce phase-coded spike trains, making them an ideal input for SNNs. Unlike event cameras or passive SPAD imagers that primarily encode spikes based on rate, the phase-coded spike input presents a greater challenge for SNNs. While one can select encoding methods and data preprocessing schemes arbitrarily on the PC, the hardware limits such flexibility. Hence, it is crucial to consider hardware feasibility when choosing the encoding method and data preprocessing scheme.

Straightforwardly, the reference and detection signals can be connected to the SNN directly, or the sum of them, as shown in Figure 2. It is worth noting that the latter is a special case of the former, as the OR gate can be modeled as a spiking neuron with the same synaptic weights for the reference and detection signals. Nevertheless, this scheme gives rise to two challenges: the ultra-long sequences and the variance in spike density.

In active time-resolved SPAD imaging, achieving higher temporal resolution is always desirable to enhance precision, resulting in an increased number of timesteps within a repetition period. Besides, hundreds to thousands of photons are typically necessary to construct a comprehensive event picture. Thus the total timesteps can easily reach the magnitude of millions. In practice, the ratio between photon rate and repetition frequency is kept low to avoid the pile-up effect, which means that there are many repetition periods where no photon is detected. These additional "blank" inputs will further lengthen the already long sequence by several orders of magnitudes. The number of total timesteps is given by

$$N = \frac{N_T \times N_C}{\phi}, \tag{1}$$

where N_T is the number of timesteps in one repetition period, N_C is the number of desired photons, and ϕ is the ratio between photon rate and repetition frequency. The ultra-long sequence results in inefficiency and excessive computation during the inference and makes it impossible to train the neural network through backpropagation through time (BPTT) due to the gradient exploding/diminishing problem.

The probability of photon reception within a single repetition period varies across pixels and cases, which leads to the variation in the number of "blank" inputs. When employing spiking neuron models with temporal dynamics such as membrane potential decay and input decay, it would be challenging for the SNN to handle inputs with such a high temporal dynamic range.

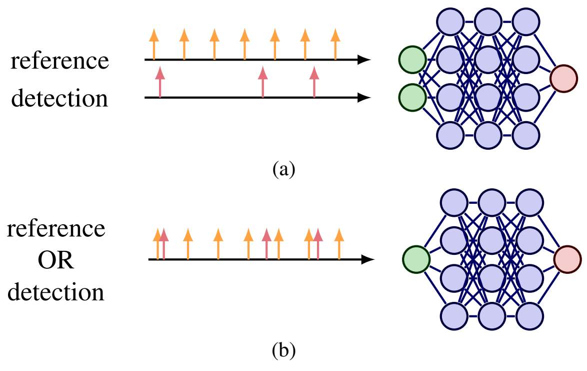


Figure 2. Intuitive input coding for single-photon detectors. (a) The SNN has two input nodes, which take the reference signal and the detection signal as input separately. (b) The SNN has only one input node, which takes the sum of the reference and detection signals as input.

In order to implement the SNN for active time-resolved SPAD imaging on hardware, it is essential to employ a hardware-feasible encoder to convert the dense-coded spike trains from SPADs to denser and more informative ones, where the "blank" inputs are eliminated and the encoded information is easier to learn. Tailored training techniques are also required for supervised learning on regression tasks.

3. Methods

3.1. Architecture

3.1.1 Neuron Model

We adopt Leaky Integrate-and-Fire (LIF) model to keep the balance of the trade-off between the simplicity and capability of the neural network. Considering that subtraction is easier than multiplication in hardware implementation, we assume that the membrane potential decays linearly instead of exponentially. The membrane potential U of neuron i in layer l at timestep n with soft reset is given by:

$$\begin{aligned} U_i^{(l)}[n] = \quad & U_i^{(l)}[n-1] - \tau_{decay} && \big\}\ \text{decay} \\ & -V_{thre} S_i^{(l)}[n-1] && \big\}\ \text{reset} \\ & + \textstyle\sum_{j=0}^{N_{l-1}} w_j^{(l-1)} S_j^{(l-1)}[n] && \big\}\ \text{input} \end{aligned} \tag{2}$$

where τ_{decay} is the decay constant, V_{thre} is the firing threshold, N_l is the number of neurons in the layer l, w is the synapse weight, and S is the output of spiking neurons. The spiking neuron fires when the membrane potential exceeds the threshold:

$$S_i^{(l)}[n] = \begin{cases} 1, & \text{if} \quad U_i^{(l)}[n] \geq V_{thre} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

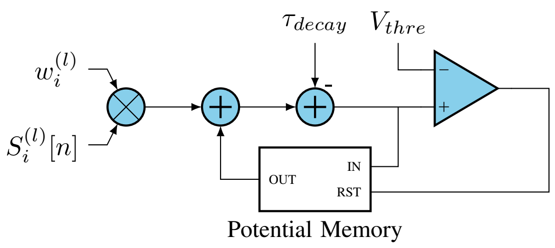


Figure 3. Schematic of the simplified leaky integrate-and-fire (LIF) spiking neuron model.

When the hard reset is adopted, the membrane potential $U_i^{(l)}$ is simply reset to 0 when it exceeds the threshold. Hard reset is used for Transporter SNN, and soft reset is used for RS SNN.

The simplified LIF model requires minimal hardware resources to implement. The necessary electronic building blocks are two adders for integration and decay, one comparator for firing detection, and one memory for membrane potential storage. The schematic is shown in Figure 3. The simple structure allows it to be implemented massively on the sensor [6].

The spiking neurons and networks are built with *SpikingJelly*[1], an open-source deep learning framework for SNN based on *PyTorch*. To realize the simplified LIF model, the perfect integrate-and-fire model (IF) `neuron.IFNode` in *SpikingJelly* is used, and a bias is added along with synaptic weights, serving as a constant current into the IF model. It is worth noting that the bias can be either positive or negative.

3.1.2 Network Topology

Considering the limited resource on the hardware and seeing this work as proof of concept, we only adopt one-hidden-layer SNNs and two-hidden-layer SNNs. More complex topologies, such as recurrence and convolution, will be studied in the future [36].

Transporter SNN uses the two-hidden-layer architecture. Both input and output layers have only one node. The first and second hidden layer has 256 and 128 neurons respectively. RS SNN uses the one-hidden-layer architecture due to its long sequence. Both input and output layers have only one node. The only hidden layer has 512 neurons.

3.1.3 Decoding and Loss Function

The potential decoders and loss functions have been well summarized in the literature [10]. This work uses a variation of mean square spike rate (MSSR) and mean square membrane (MSM). The mean of the output along timesteps

[1]https://github.com/fangwei123456/spikingjelly/tree/master

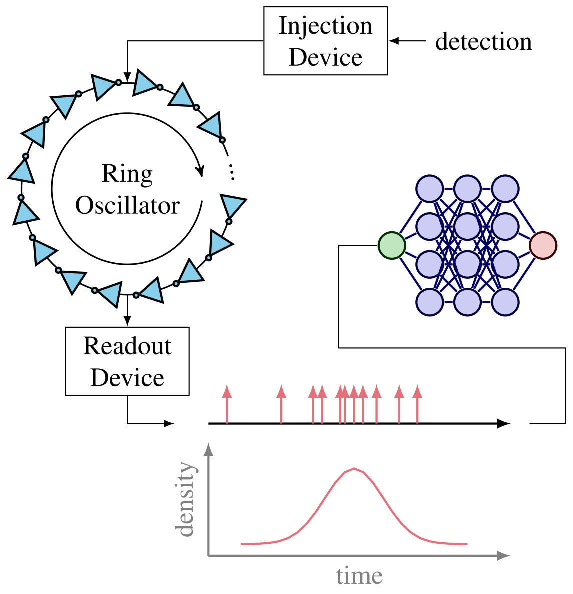


Figure 4. The proposed Transporter SNN model. The detected spikes are injected into an RO, whose period equals the repetition period of the laser. When the RO is synchronized with the laser, the incoming arrival times are folded into one spike sequence, which is read out after the acquisition. The density of sequence is basically proportional to the histogram, which is an ideal input for SNNs.

is interpreted as the prediction:

$$\hat{y} = \frac{1}{N_T} \sum_{n=0}^{N_T} y[n] \tag{4}$$

The output node can be modeled as an IF neuron with an infinite threshold and no leakage, where the prediction $\hat{y}$ is proportional to the membrane potential when N_T is fixed.

Mean absolute percentage error (MAPE) is used as the loss function for the experiment, which is defined as

$$\mathcal{L}_{MAPE} = \text{MAPE}(y, \hat{y}) = \frac{1}{N_B} \sum_{i}^{N_B} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{5}$$

where N_B is the batch size and y_i is the label of the sample i. However, common loss functions such as $L1$ loss and $L2$ loss can be used as well.

3.2. Transporter SNN

3.2.1 Hardware Structure and Encoding

After Captain Montgomery Scott experienced a crash at Dyson sphere, he rigged the Transporter and left himself in the diagnostic loop indefinitely until a rescue vessel could come and "rematerialize" him (*Star Trek* S6E4). Inspired by this "Transporter suspension", we propose Transporter SNN, creating a loop to store temporal information in time instead of memory until "rematerialization" for processing.

The schematic of Transporter SNN is illustrated in Figure 4. The spike generated by a SPAD is injected into a ring oscillator (RO), with delay elements of, if any, a few ps. The injected spikes would start circulating indefinitely until "rematerialization" by a readout circuit. The RO is supposed to synchronize with the reference signal, keeping the same period. After running for thousands of periods until the required information is collected, the spikes are read out sequentially, where the differences among arrival times are maintained. The resulting sequence is exactly the binarized sum of all repetition periods, thus the density of the spike through time is basically the histogram of arrival times.

The number of timesteps is reduced by several orders of magnitude, significantly improving the efficiency and facilitating the training. Since the delay element can only store binary information, several spikes falling into the same delay element can cause information loss and distortion. Thus one has to balance the trade-off between hardware implementation difficulty and information fidelity in practice.

3.2.2 Training

Transporter SNN is trained with Surrogate Gradient [20, 36]. With the folding of the time domain by the RO, millions of timesteps of the sequence are compressed into thousands to tens of thousands of timesteps, which makes it possible to train the SNN with BPTT.

Surrogate Gradient smoothes the SNN and helps the gradients "flow" backward during BPTT. The firing function (Eq. 3) is non-differentiable at $U = V_{thre}$ and has derivatives of 0s elsewhere, which makes it impossible to use BPTT directly. To overcome this, the non-differentiable firing function is replaced by a differentiable surrogate function, e.g. Sigmoid and arctan functions, in the backward path. Arctan is used here for the training:

$$S(U) = \frac{1}{\pi} \arctan\left(\frac{\pi}{2}\alpha(U - V_{thre})\right) + \frac{1}{2} \tag{6}$$

where α is a scaler. Its derivative is given by:

$$\frac{\partial S}{\partial U} = \frac{\alpha}{2\left(1 + \left(\frac{\pi}{2}\left(U - V_thre\right)\right)^2\right)} \tag{7}$$

which is everywhere defined, continuous, and non-zero. $\partial S/\partial U$ reaches the maximum at $U = V_{thre}$.

3.3. Reversed Start-stop SNN

3.3.1 Hardware Structure and Encoding

Inspired by the concept of reversed start-stop TDCs [38], we apply a similar mechanism to SNNs in order to eliminate "blank" repetition periods, named Reversed Start-stop SNN. The schematic is illustrated in Figure 5. The fundamental idea behind this approach is to pause the SNN's internal clock when no photons are detected and resume it when photons arrive. Since a detector can only determine if a photon arrives during a repetition period or at its end, the SNN is designed to be active only when photons are detected. Unlike conventional timers that are triggered by a reference signal and halted by an event signal, in this case, the SNN is enabled by the event signal and disabled by the reference signal. Within the SNN, the clock keeps ticking, resulting in an exact interspike-interval-coded representation of the incoming spike train. The interval is determined by $T - t$, where T represents the repetition period and t denotes the arrival time of the photons. To implement this mechanism in hardware, a Flip-Flop can be employed. The detection signal sets the Flip-Flop, while the reference signal resets it. The output of the Flip-Flop is then connected to the EN pin of the SNN. As a result, all empty repetition periods are disregarded, and the number of timesteps required for processing a single photon is reduced.

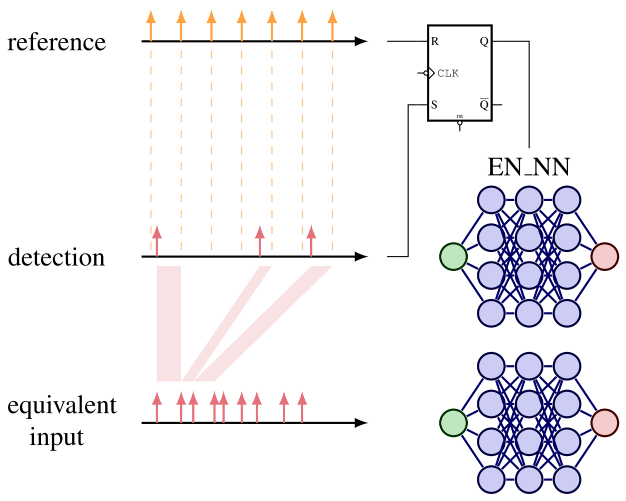


Figure 5. The proposed Reversed Start-stop (RS) SNN model. The reference and detection signals are connected to an SR flip-flop, where the flip-flop is set by the detection signal and reset by the reference signal. The output of the flip-flop is used to enable/disable the SNN. The SNN takes the detection signal as the only input. The equivalent model is shown below. As for the SNN, it is equivalent to taking an ISI-coded spike train continuously.

While neurobiologists are debating on the neural coding scheme that the brain adopts, ISI coding is often neglected [3]. This coding scheme, however, might carry more information than rate coding and phase coding [16]. The temporal information in the phase coding is preserved after the conversion to ISI coding. The spike rate of the ISI-coded spike trains is linear with the inverse of the average spiking time, i.e. the average arrival time of the photons for SPAD:

$$\bar{t} = T - \frac{1}{\frac{N_C}{N}} \times \text{LSB} \tag{8}$$

where T is the repetition period and LSB is the least-significant bit (temporal resolution). $\bar{t}$ is an informative parameter for most active SPAD imaging tasks, which will facilitate the training of the SNN, as described in Section 3.3.2.

3.3.2 Training

Training ISI-coded SNNs poses greater challenges than the density-coded ones, primarily due to their ultra-long sequence. Training with Surrogate Gradient directly would be excessively time-consuming and computationally intensive, and would suffer from gradient exploding/diminishing problems. To address these issues, we adopt a two-step training strategy for RS SNN. We start with training an ANN, which takes the spike rate as input and makes predictions, then it is converted to an SNN. This initial training phase is referred to as the "warm-up" training. After that, the converted SNN is trained directly on the spike dataset with Surrogate Gradient.

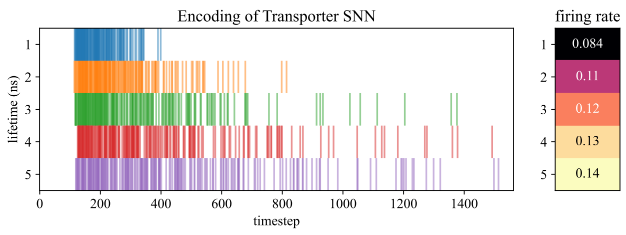
An ANN of the same topology as the desired SNN is built first, whose input is the average of the spike train (i.e. the spiking rate), the output is the label, and the activation function is Rectified linear unit (ReLU). The ANN is trained until convergence. Owing to the strong relationship between the ReLU and the firing rate of IF neurons with the soft reset, the ANN can be converted to the SNN [29]. The conversion is realized by the `ann2snn` module in *SpikingJelly*. It is worth noting that the `VoltageScaler` created by the `ann2snn` module can be incorporated into the weights and biases.

The converted SNN is then trained directly with Surrogate Gradient, as described in Section 3.2.2. It is expected that the performance can be improved by learning the temporal information embedded in the ISI-coded spike trains.

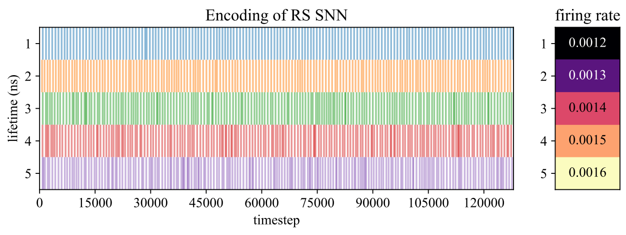
4. Experiments

4.1. Data Acquisition

Synthetic datasets are created for training and evaluation. The modeling of fluorescence is described in [17]. Fluorescence, instrument response, background noise, and dark counts are taken into account. Timestamps are generated through the Monte Carlo simulation and are transformed into spike trains. The fluorescence lifetime ranges from 1 ns to 6 ns. Three background noise (including dark counts) levels are considered, which are 0, 1%, and 5%. The center of the instrument response is at 1.968 ns. Other parameters are kept the same as in [17].



(a) Examples of encoding of Transporter SNNs. There are 256 repetition periods and 1563 timesteps for each repetition period.



(b) Examples of encoding of RS SNNs. There are 128 repetition periods and 1000 timesteps for each repetition period.

Figure 6. Examples of the encoding of the proposed SNNs. Spike trains for fluorescence TCSPC data with different lifetimes are shown here. The LSB is 0.016 ns and no background noise is considered. The shift and FWHM of the IRF are 1.968 ns and 0.1673 ns. For better visualization, different numbers of repetitions periods and timesteps are used.

The timestamps are transformed into density-coded and ISI-coded spike trains for Transporter SNN and RS SNN respectively. Examples of these spike trains are illustrated in Figure 6. For density-coded spike trains, it is assumed that the temporal resolution is 0.05 ns, the repetition period is 50 ns (equivalent to a 20 MHz laser), and 256 spikes are found in each sequence. For ISI-coded spike trains, it is assumed that the temporal resolution is 0.05 ns, the repetition period is 12.5 ns (equivalent to an 80 MHz laser), and the total number of timesteps for one sequence is 50,000.

Experimental data from [17] is utilized to evaluate the performance of SNNs on real-world data, which contains the TCSPC data of a fluorescence lifetime-encoded beads sample. Random distributions are generated from the histograms, from which the timestamps are sampled to ensure that it has the same form as the training set.

4.2. Training

4.2.1 Transporter SNN

Transporter SNN is trained with Surrogate Gradient. Arctan with $\alpha = 2$ is used as the surrogate function. Since we are training on synthetic data, the training set can be generated for each batch to avoid overfitting. The batch size is 32 and the SNN is trained on 200,000 batches. Adam optimizer is used with an initial learning rate of 0.001, which is halved

Models	MAPE		
	No Noise	1% Noise	5% Noise
CMM	0.0507	0.0838	0.3455
CMM[†]	0.0507	0.0723	0.1410
LSTM	0.0485	0.0503	0.0555
Transporter SNN	0.0542	0.0591	0.0633
ANN[††]	0.0602	0.0622	0.0666
Converted RS SNN[††]	0.0603	0.0629	0.0674
RS SNN[††]	0.0601	0.0610	0.0655

Table 1. Comparison of performance of the proposed models and benchmarks on fluorescence lifetime estimation. (MAPE) is adopted as the metric. Three levels of background noise are considered. The LSB is 0.05 ns. There are 256 repetitions and 1000 timesteps for each repetition.

[†] CMM with background subtraction, assuming that the background level is known.

[††] 250 timesteps are considered for each repetition period, which is equivalent to a 12.5 ns repetition period. In total, there are 50,000 timesteps for each sequence. A sequence contains 323.6 photons on average. The longer the lifetime, the more photons in the sequence.

every 10,000 batches. The training takes 38 min on our workstation (AMD Ryzen Threadripper PRO 3945WX and NVIDIA RTX A4500).

4.2.2 RS SNN

An ANN is initially constructed and trained, and subsequently converted into an SNN. To maintain the same topology as of the SNN, an ANN with a single hidden layer is created, featuring 512 hidden neurons activated by the ReLU function. The training process involves 50,000 batches, each comprising 128 samples. The parameters are updated using the Adam optimizer with a learning rate of 0.001. The weights acquired during training are directly utilized as the synaptic weights for the SNN. Furthermore, two `VoltageScaler` components are learned from a smaller dataset consisting of 1,000 samples, with one placed before the hidden layer and the other after it.

Following the conversion process, the SNN undergoes training using Surrogate Gradient. The training procedure is similar to that employed for the Transporter SNN, with the exception of utilizing 250,000 batches. As a result of the long ISI-coded sequence, the training process takes 2 hours to complete.

4.3. Evaluation on Synthetic Data

Transporter SNN and RS SNN, as well as benchmarks including Center-of-Mass Method (CMM) [15] and Long Short-Term Memory (LSTM) [17], are evaluated on the synthetic data. All the methods are tested on a dataset comprising 100,000 samples. MAPE (Eq 5) is adopted as the metric. The result is shown in Table 1.

The CMM, although initially effective in noise-free environments, experiences a significant decline in performance when faced with even minor background noise, even when utilizing background subtraction. LSTM demonstrates superior performance across all scenarios. Despite its inferior performance compared to the CMM in noise-free conditions, the Transporter SNN surpasses the CMM by a substantial margin when confronted with background noise. It is important to highlight that the Transporter SNN possesses a remarkably simple structure, and its performance is expected to be enhanced by using more complex topologies.

Owing to its distinct coding scheme, the RS SNN is evaluated on a different basis. Nevertheless, the ANN is supposed to perform slightly better than CMM with background subtraction since there is a simple relationship between the CMM and the number of spikes. The RS SNN converted from the ANN observes a slight drop in accuracy, but it is restored and improved through BPTT with Surrogate Gradient. Hence we believe that temporal information is learned through the training process. The performance of RS SNN is expected to be improved by adopting more complex topologies as well.

Figure 7 illustrates the firing of neurons in the Transporter SNN over time. From timestep 50 when the input spikes come in, a high rate of firing is observed among some neurons (e.g. `Neuron` 15, 24, 34, and 47), exhibiting excitatory behavior, and a decrease of firing rate is also observed among some neurons (e.g. `Neuron` 5 and 13), exhibiting inhibitory behavior. These behaviors are regulated by the sign and amplitude of the synaptic weights and decay constant. "Dead neurons", which do not fire a single spike during the whole process, are found as well (e.g. `Neuron` 0, 11, 20, and 32). These neurons could be pruned during the inference to reduce hardware resources and computational complexity.

4.4. Evaluation on Experimental Data

Transporter SNN and RS SNN, which were exclusively trained on synthetic data, are tested using real-world data. In this benchmark, CMM serves as the reference, with the Least-square (LS) fitting considered as the gold standard. The result is illustrated in Figure 8. The sample consists of fluorescent beads with three distinct lifetimes, evident from the three peaks present in all the histograms. The LS Fitting histogram displays low-variance Gaussian profiles, which is expected due to the enhanced statistical accuracy resulting from a higher number of photons. The histogram profiles of Transporter SNN, RS SNN, and CMM exhibit similar characteristics, indicating that the proposed SNNs perform effectively in real-world scenarios.

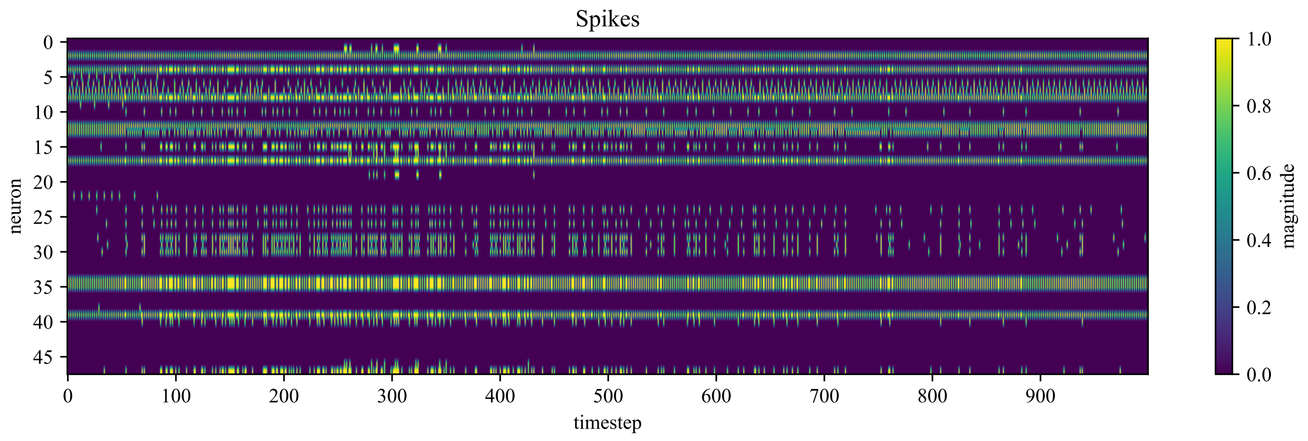


Figure 7. Heatmap of the firing of spiking neurons. The first 48 neurons of the one hidden layer Transporter SNN are shown here.

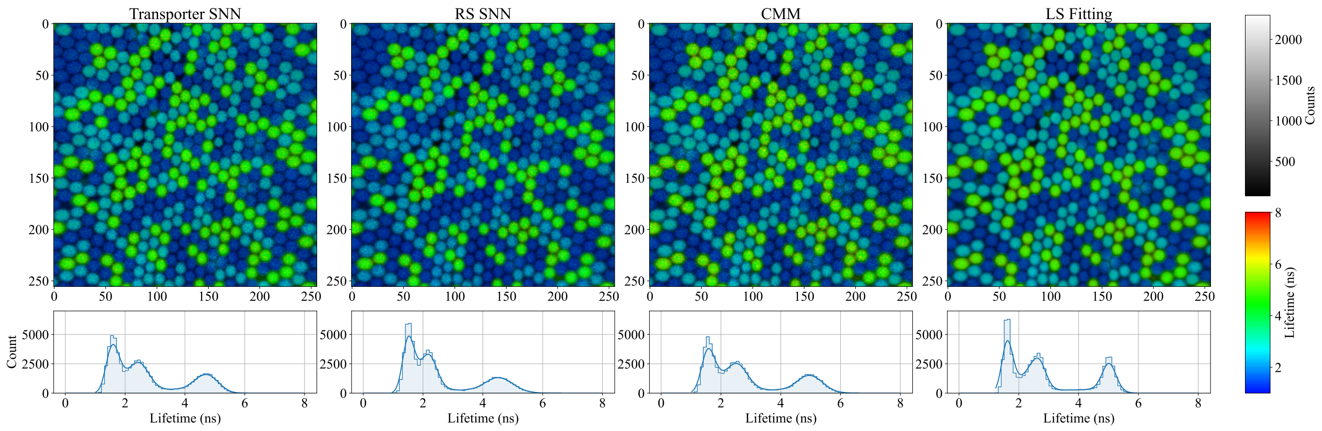


Figure 8. Comparison of Transporter RNN, RS RNN, CMM, and LS fitting on experimental data. The data are downsampled for Transporter SNN, RS SNN, and CMM. 256 photons with 0.05 ns LSB are used for Transporter SNN and CMM. 50,000 timesteps, 0.05 ns LSB, and 12.5 ns repetition period are used for RS SNN. All the available data are used for LS fitting, which serves as the gold standard here.

5. Conclusion and Outlook

In this paper, we present two SNN frameworks, Transporter SNN and RS SNN, for active time-resolved SPAD imaging, detailing the simplified LIF neuron model and its hardware, the encoder and its hardware, and the tailored training scheme. We emphasize the necessity of employing dedicated encoder hardware to convert phase-coded spike trains from SPADs to denser and more informative spike trains for more efficient training and inference, and also eliminate the TDC on the hardware level to release resources. The performance of the proposed SNNs is demonstrated through testing on synthetic and experimental data, exhibiting satisfactory results even with simple topologies such as one-hidden-layer and two-hidden-layer neural networks. We hope that this work can serve as a foundational reference for the broader adoption of SNNs and inspire the research community to develop more advanced SNN-based models for active time-resolved SPAD imaging applications.

The following studies could be carried out on both hardware and software levels. On the hardware level, the dedicated encoder and the SNN are going to be implemented on the FPGA and further on the chip to realize near-sensor and in-sensor processing, which could reduce power consumption and latency and allow high throughput processing. 3D stacking technology is envisioned to further improve the performance, making it possible to be migrated to portable devices. On the software level, more complex network topologies are going to be explored to improve the performance of SNNs, and they will be further trained for high-level downstream tasks such as object detection and image segmentation.

References

[1] Apple unveils new ipad pro with breakthrough lidar scanner and brings trackpad support to ipados. *Apple*, 3/18/2020. 1

[2] Introducing apple vision pro: Apple's first spatial computer. 6/5/2023. 1

[3] Daniel Auge, Julian Hille, Etienne Mueller, and Alois Knoll. A survey of encoding techniques for signal processing in spiking neural networks. *Neural Processing Letters*, 53(6):4693–4710, 2021. 5

[4] Sander M. Bohte, Joost N. Kok, and Han La Poutré. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002. 2

[5] Olaf Booij and Hieu tat Nguyen. A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters*, 95(6):552–558, 2005. 2

[6] Maxence Bouvier, Alexandre Valentian, Thomas Mesquida, Francois Rummens, Marina Reyboz, Elisa Vianello, and Edith Beigne. Spiking neural networks hardware implementations and challenges. *ACM Journal on Emerging Technologies in Computing Systems*, 15(2):1–35, 2019. 4

[7] Claudio Bruschini, Harald Homulle, Ivan Michel Antolovic, Samuel Burri, and Edoardo Charbon. Single-photon avalanche diode imagers in biophotonics: review and outlook. *Light: Science & Applications*, 8(1):87, 2019. 1

[8] Canon. Canon developing world-first ultra-high-sensitivity ilc equipped with spad sensor, supporting precise monitoring through clear color image capture of subjects several km away, even in darkness - canon press centre. 4/3/2023. 1

[9] Jianhao Ding, Zhaofei Yu, Yonghong Tian, and Tiejun Huang. Optimal ANN-SNN conversion for fast and accurate inference in deep spiking neural networks. 2

[10] Jason K. Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D. Lu. Training spiking neural networks using lessons from deep learning. *arXiv preprint arXiv:2109.12894*, 2021. 2, 4

[11] Mathias Gehrig, Sumit Bam Shrestha, Daniel Mouritzen, and Davide Scaramuzza. Event-based angular velocity regression with spiking networks. *IEEE International Conference on Robotics and Automation (ICRA)*. 2

[12] Jesse Hagenaars, Federico Paredes-Valles, and Guido de Croon. Self-supervised learning of event-based optical flow with spiking neural networks. *Advances in Neural Information Processing Systems*, 34:7167–7179, 2021. 2

[13] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: Spiking neural network for energy-efficient object detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):11270–11277, 2020. 2

[14] Eimantas Ledinauskas, Julius Ruseckas, Alfonsas Juršėnas, and Giedrius Buračas. Training deep spiking neural networks. 2

[15] Day-Uei Li, Eleanor Bonnist, David Renshaw, and Robert Henderson. On-chip, time-correlated, fluorescence lifetime extraction algorithms and error analysis. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 25(5):1190–1198, 2008. 7

[16] Meng Li and Joe Z. Tsien. Neural code-neural self-information theory on how cell-assembly code rises from spike time and neuronal variability. *Frontiers in cellular neuroscience*, 11:236, 2017. 5

[17] Yang Lin, Paul Mos, Andrei Ardelean, Claudio Bruschini, and Edoardo Charbon. Recurrent neural network-coupled SPAD TCSPC system for real-time fluorescence lifetime imaging. *arXiv preprint arXiv:2306.15599*, 2023. 6, 7

[18] Riccardo Massa, Alberto Marchisio, Maurizio Martina, and Muhammad Shafique. An efficient spiking neural network for recognizing gestures with a dvs camera on the loihi neuromorphic processor. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, uuuu-uuuu. 2

[19] Sambit Mohapatra, Thomas Mesquida, Mona Hodaei, Senthil Yogamani, Heinrich Gotzig, and Patrick Mader. SpikiLi: A spiking simulation of LiDAR based real-time object detection for autonomous driving. 2

[20] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. 2, 5

[21] Paul Kirkland, Valentin Kapitany, Ashley Lyons, John Soraghan, Alex Turpin, Daniele Faccio, and Gaetano Di Caterina. Imaging from temporal data via spiking convolutional neural networks. pages 66–85. SPIE, 2020. 2

[22] Nicolas Perez-Nieves and Dan Goodman. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems*, 34:11795–11808, 2021. 2

[23] Nicolas Perez-Nieves and Dan F. M. Goodman. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems (NeurIPS*, 2021. 2

[24] Michael Pfeiffer and Thomas Pfeil. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, 12:774, 2018. 2

[25] Wachirawit Ponghiran and Kaushik Roy. Spiking neural networks with improved inherent recurrence dynamics for sequential learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):8001–8008, 2022. 2

[26] I. Prochazka, K. Hamal, and B. Sopko. Recent achievements in single photon detectors and their applications. *Journal of Modern Optics*, 51(9-10):1289–1313, 2004. 1

[27] Ulysse Rançon, Javier Cuadrado-Anibarro, Benoit R. Cottereau, and Timothée Masquelier. StereoSpike: Depth Learning with a Spiking Neural Network. 2

[28] A. Rochas, M. Gani, B. Furrer, P. A. Besse, R. S. Popovic, G. Ribordy, and N. Gisin. Single photon detector fabricated in a complementary metal–oxide–semiconductor high-voltage technology. *Review of Scientific Instruments*, 74(7):3263–3270, 2003. 1

[29] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017. 2, 6

[30] Mst Shamim Ara Shawkat, Md Musabbir Adnan, Rocco D. Febbo, John J. Murray, and Garrett S. Rose. A single chip

spad based vision sensing system with integrated memristive spiking neuromorphic processing. *IEEE Access*, 11:19441–19457, 2023. 2

[31] Jason T. Smith, Alena Rudkouskaya, Shan Gao, Juhi M. Gupta, Arin Ulku, Claudio Bruschini, Edoardo Charbon, Shimon Weiss, Margarida Barroso, Xavier Intes, and Xavier Michalet. In vitro and in vivo NIR fluorescence lifetime imaging with a time-gated SPAD camera. *Optica*, 9(5):532–544, 2022. 1

[32] Sony Semiconductor Solutions Group. Sony semiconductor solutions to release spad depth sensor for smartphones with high-accuracy, low-power distance measurement performance, powered by the industry's highest photon detection efficiency, 6/25/2023. 1

[33] Federica Villa, Fabio Severini, Francesca Madonini, and Franco Zappa. SPADs and SiPMs arrays for long-range high-speed light detection and ranging (LiDAR). *Sensors (Basel, Switzerland)*, 21(11):3839, 2021. 1

[34] Carlos D. Virgilio G, Juan H. Sossa A, Javier M. Antelis, and Luis E. Falcón. Spiking neural networks applied to the classification of motor tasks in EEG signals. *Neural networks : the official journal of the International Neural Network Society*, 122:130–143, 2020. 2

[35] Wei Wang, Shibo Zhou, Jingxi Li, Xiaohua Li, Junsong Yuan, and Zhanpeng Jin. Temporal pulses driven spiking neural network for time and power efficient object recognition in autonomous driving. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6359–6366. IEEE, 2021. 2

[36] Bojian Yin, Federico Corradi, and Sander M. Bohté. Effective and efficient computation with multiple-timescale spiking recurrent neural networks. In *International Conference on Neuromorphic Systems 2020*, pages 1–8, New York, NY, USA, 2020. ACM. 4, 5

[37] F. Zappa, S. Tisa, A. Tosi, and S. Cova. Principles and features of single-photon avalanche diode arrays. *Sensors and Actuators A: Physical*, 140(1):103–112, 2007. 1

[38] Chao Zhang, Scott Lindner, Ivan Michel Antolovic, Martin Wolf, and Edoardo Charbon. A CMOS SPAD Imager with Collision Detection and 128 Dynamically Reallocating TDCs for Single-Photon Counting and 3D Time-of-Flight Imaging. *Sensors (Basel, Switzerland)*, 18(11):4016, 2018. 5