Foveal-pit inspired filtering of DVS spike response

1st Shriya T.P. Gupta
Microsoft Research and Development Center
Microsoft Corporation Pvt. Ltd.
Bangalore, India
shriyatp99@gmail.com

2nd Pablo Linares-Serrano
Instituto de Microelectrónica de Sevilla
IMSE-CNM (CSIC and Universidad de Sevilla)
Sevilla, Spain
pablolinareserrano@gmail.com

3rd Basabdatta Sen Bhattacharya
Department of Computer Science
BITS Pilani Goa Campus
Goa, India
basabdattab@goa.bits-pilani.ac.in

4th Teresa Serrano-Gotarredona
Instituto de Microelectrónica de Sevilla
IMSE-CNM (CSIC and Universidad de Sevilla)
Sevilla, Spain
terese@imse-cnm.csic.es

***Abstract*—In this paper, we present results of processing Dynamic Vision Sensor (DVS) recordings of visual patterns with a retinal model based on foveal-pit inspired Difference of Gaussian (DoG) filters. A DVS sensor was stimulated with varying number of vertical white and black bars of different spatial frequencies moving horizontally at a constant velocity. The output spikes generated by the DVS sensor were applied as input to a set of DoG filters inspired by the receptive field structure of the primate visual pathway. In particular, these filters mimic the receptive fields of the midget and parasol ganglion cells (spiking neurons of the retina) that sub-serve the photo-receptors of the foveal-pit. The features extracted with the foveal-pit model are used for further classification using a spiking convolutional neural network trained with a backpropagation variant adapted for spiking neural networks.**

***Index Terms*—dynamic vision sensor, neural filtering, spiking neural network, classification, difference of gaussian, convolution, foveal-pit**

I. Introduction

Recent advances in deep learning [1], [2] have led to state-of-the-art performance for varied classification tasks in natural language processing, computer vision and speech recognition. Traditional Artificial Neural Networks (ANN) use idealized computing units which have a differentiable, non-linear activation function allowing stacking of such neurons in multiple trainable layers. The existence of derivatives makes it possible to carry out large scale training of these architectures with gradient based optimization methods [3] using high computing resources like Graphic Processing Units (GPU). However, this prevents the use of such deep learning models for essential real-life applications like mobile devices and autonomous systems that have limited compute power.

Spiking Neural Networks (SNN) have been proposed as an energy-efficient alternative to ANNs as they simulate the event-based information processing of the brain [4]. These bio-inspired SNNs follow an asynchronous method of event processing using spiking neurons. The internal state of a spiking neuron is updated when it receives an action potential and consequently an output spike is fired when the membrane voltage crosses a pre-defined threshold. Further, improvements in neuromorphic engineering allow the implementation of SNNs on neuromorphic hardware platforms [5] that lead to a much higher efficiency in terms of power and speed compared to conventional GPU based computing systems.

Although SNNs are considered as the third generation of neural networks holding the potential for sparse and low-power computation, their classification performance is considerably lower than those of ANNs. This can be attributed to the fact that gradient optimization techniques like the backpropagation algorithm can't be implemented in SNNs due to the discrete nature of spiking neurons. A common technique for training SNN models is the Hebbian learning inspired Spike Timing Dependent Plasticity (STDP) that is used in several state-of-the-art approaches [6], [7]. Other works like [8], [9] have adapted the gradient descent algorithm for SNNs using a differentiable approximation of spiking neurons. Our approach also employs a similar modified backpropagation algorithm proposed by Hunsberger et al. [10] that is implemented in the Nengo-DL library [11].

As shown by Camunas-Mesa et al. [12], the efficiency gain of SNNs from event-based processing can be further improved through the use of inputs from event-based sensors like a neuromorphic Dynamic Vision Sensor (DVS) [13]. Event driven sensors represent the information dynamically by asynchronously transmitting the address event of each pixel and hence avoid processing redundant data. However, the classification accuracy drops drastically when using real sensory data from a physical spiking silicon retina, since the spike events are no longer Poissonian [14].

In our previous work [15], we had demonstrated the effect of foveal-pit inspired filtering for synthetically generated datasets like MNIST [3] and Caltech [16]. In this work, we present the

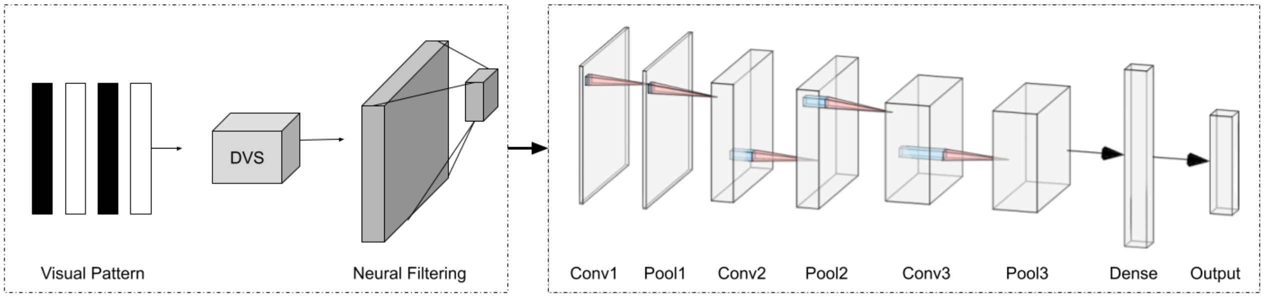


Fig. 1. The two-stage architecture of the proposed DVS based spiking convolutional neural network. In the first stage, the DVS is stimulated with a pattern of vertical white and black bars and the generated spike responses are processed using foveal-pit inspired DoG filters. The second stage is composed of the convolutional and pooling layers which are used to classify the features extracted from the first stage.

results of applying similar neural filtering to data generated by the DVS. In our proposed model, we process DVS outputs using bio-inspired filters that simulate receptive fields of the midget and parasol ganglion cells of the primate retina. The DVS is stimulated with vertical black and white bars having a constant displacement of 2 pixels from frame to frame. The foveal-pit informed Difference of Gaussian (DoG) filters are applied to the DVS recordings in order to capture the most perceptually important information from the input data. The use of DoG functions to model retinal filters was originally proposed by Rullen et al. [17] and the receptive fields of the foveal-pit are implemented as in Bhattacharya et al. [18].

The processed features are then used to perform the classification using a Spiking Convolutional Neural Network (SCNN). The SCNN architecture is inspired by two previous works viz. Diehl et al. [7] and Kheradpisheh et al. [6], while the model is implemented as in Gupta et al. [15]. Each input is presented to the network for a total duration of 60 timesteps and the predictions are assigned based on the voltages measured from the output neurons. The empirical results demonstrate that the application of neural filtering to DVS recordings leads to an improvement of 35% in classification accuracy compared to the unfiltered DVS spike responses. Out of the filtered scenarios, the highest performance of 100% is achieved using the off-center parasol ganglion cells.

The rest of the paper is organized as follows: Section II describes the architecture of the proposed model including the response generation and filtering, Section III provides the results of the experiments and Section IV contains the conclusion and future directions.

II. Methodology

The overall architecture of our model consists of two main stages: the first stage is made up of the DVS response generation and neural filtering of output spikes; the second stage consists of performing classification using the SCNN. The proposed model is shown in Fig. 1 and each of the individual stages are covered in detail in the following subsections.

A. *Dynamic Vision Sensor Responses*

We have used a 128 × 128 sized neuromorphic DVS developed by Serrano-Gotarredona et al. [13] to capture non-synthetic visual data. Each pixel of the DVS processes the input continuously and emits a spike based on the variation in the illumination impinging upon it [19]. A sample illustration is provided in Fig. 2 using a sinusoidal input stimulus having a frequency of 10Hz. The first row depicts the pixel's illumination over time whereas the remaining two rows capture the emission of spikes over the same duration corresponding to changes in illumination. An increase in illumination leads to a positive spike whereas a decrease in illumination leads to a negative spike as seen in the last row of Fig. 2.

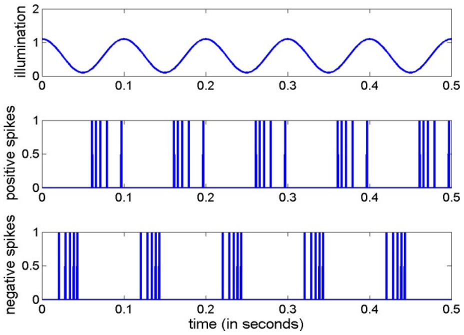


Fig. 2. Illustration of the DVS spike response generated using a sinusoidal input stimulus. The first row represents the pixel's illumination over time. The second row depicts the positive spikes whereas the last row represents the negative spikes corresponding to a decrease in illumination [19].

For our experiments, the DVS was placed in front of a monitor displaying a pattern of equally wide black and white vertical bars as shown in Fig. 3. The bars were moved horizontally across the screen such that a displacement of 2 pixels is applied from frame to frame. The number of bars were varied from 2, 4, 8, 16, 32, 64 to 128 and these $K = 7$ categories correspond to the final labels for our multiclass classification problem. The events generated by the DVS were captured in the Address Event Representation (AER) format using the jAER software [20].

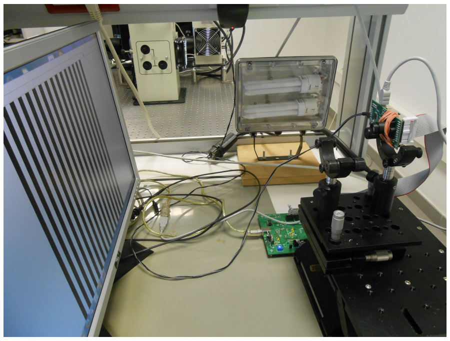


Fig. 3. The DVS setup to record spike responses when presented with a simple pattern moving across a computer screen as visual stimulus.

B. Retina-inspired filtering

The DVS recordings generated from the first stage of our model are passed to a set of neural filters simulating the primate visual system. As proposed by Kheradpisheh et al. [6], we have used DoG functions to implement these biologically inspired filters sub-serving the retinal foveal pit. The foveal pit is a circular region of 200 μm diameter that lies at the center of the foveola. This region also has the highest visual acuity in the primate retina and is most accessible to incoming light. The fovea is sub-served by a ganglion cell layer composed of midget and parasol cells. The retinal ganglion cells are the only spiking neurons of the primate visual system and their axons transmit the received information from the retina to other parts of the brain.

Algorithm 1: Algorithm for filtering the DVS spike response using foveal-pit inspired DoG functions.

```
kernel = dog_func(mat_dim, cent_dev, circ_shift)
[ma, na] = size(input)
[mb, nb] = size(kernel)
filt_out = zeros(ma, na)
r1 = ceil(mb/2)
s1 = ceil(nb/2)
for i = 1 to i = ma do
    for j =1 to j = na do
        i1 = max(0, i-r1);
        for r = max(1, r1-i+1) to r = mb do
            i1 = i1 + 1
            j1 = max(0, j-s1)
            for s = max(1, s1-j+1) to s = nb do
                j1 = j1 + 1
                filt_out(i, j) += kernel(r, s) * input(i1, j1)
            end for
        end for
    end for
end for
return  filt_out
```

The midget and parasol ganglion cells have two types of centre surround receptive fields — on-centre-off-surround and off-centre-on-surround. We have modelled these receptive fields using DoG functions as specified in Bhattacharya et al. [18]. The off-center midget cells have a matrix size of 5×5 with standard deviation of 0.8 whereas the on-center midget cells are of size 11×11 with standard deviation of 1.04. Similarly, the off-center parasol cells have a size of 61×61 with a standard deviation of 8 while the on-center parasol cells are of size 243×243 with a standard deviation of 10.4. These DoG functions are then applied to the DVS spike responses using Algorithm 1.

C. Convolutional Network Architecture

The asynchronous DVS recordings generated from the previous stage are split into individual frames for training our frame-based classifier composed of convolutional layers. This modified dataset is created following the procedure of Stromatias et al. [21] to produce an analog vector representation. The SCNN architecture used in our work consists of three convolutional and pooling layers which are made up of Leaky Integrate and Fire (LIF) neurons.

TABLE I
DIMENSIONS OF THE SCNN LAYERS.

Layer	No. of filters	Input size	Kernel size
Conv1	8	(128, 128)	3
Pool1	-	(128, 128)	2
Conv2	16	(64, 64)	3
Pool2	-	(64, 64)	2
Conv3	32	(32, 32)	3
Pool3	-	(32, 32)	2
Flatten	-	(16, 16)	-
Dense	-	(1, 8192)	-
Outputs	-	(1, 7)	-

Traditional deep learning architectures use sigmoid neurons which are differentiable non-linearities, whereas the spiking neurons used in SCNNs are non-differentiable. Hence, we use a differentiable approximation of the spiking neurons during training and the actual spiking neurons during inference as proposed by Hunsberger et al. [10]. Since we use a rate-based approximation during training, the model is run only for a single timestep whereas during testing with the spiking neurons, the model is run for 60 timesteps to collect the cumulative spike output over time.

The convolution is carried out on the 128×128 input arrays using filters of size 3×3. The first, second and third convolutional layers of the SCNN are made up of 2^3, 2^4 and 2^5 filters respectively, followed by a pooling operation after each convolution. The synaptic connections between the neurons of these layers are modelled as the trainable weights of the network which are optimized by minimizing the loss function of the overall SCNN. The exact dimensions of the individual layers are provided in Table I.

D. Training and Inference

For our multiclass classification problem with $K = 7$ categories, we convert the outputs of the last pooling layer into a 1-D vector using a flatten operation. This is followed by a dense layer with all-to-all connectivity having K neurons which generates a $K \times 1$ output vector. A softmax classifier is used to transform these output values into a set of K probabilities:

$$\mathbf{Y}(X, W) = \frac{e^{w_i x_i}}{\sum_{j=1}^{K} e^{w_j x_j}} \forall i = 1, \cdots, K \tag{1}$$

where $x_i \in X$ and $w_i \in W$ are the inputs and weights of the dense layer respectively, and $\mathbf{Y}$ is the prediction probabilities that sum to 1. The Negative Log Likelihood (NLL) loss for the overall network is computed using the one-hot encoded output labels $\mathbf{L}$ and the softmaxed probabilities $\mathbf{Y}$ with NLL defined as:

$$\mathbf{O}(X, W) = -\frac{1}{M} \sum_{i}^{M} \sum_{j}^{K} F_i(j) * \log(\mathbf{Y}(X, W)) \tag{2}$$

where M is the mini-batch size, $F_i(j) = 1$ when $j = L_i$ and zero otherwise. The SCNN is trained end-to-end using a spiking approximation of the backpropogation algorithm adapted for SNNs. This is done by minimizing the NLL loss using the procedure described in Gupta et al. [15] with a duration of 3 epochs and a mini-batch size of 20.

For the inference stage, we pass the input images from the testing corpora and measure the voltages (mV) of the output layer neurons. These values are generated using the probe function of the Nengo-DL library and represent the progressively increasing membrane potentials. Thus, the neuron having the highest voltage over a 60 ms simulation time period is assigned as the predicted class for that epoch.

III. Experimental Methods and Results

The filtered spikes responses from the 128×128 sized DVS sensor was split into individual frames for each recording to be passed as input to the subsequent convolutional network. This resulted in a total of 3552 images for the unfiltered scenario and a collection of 3503 images for the filtered recordings. In each case, the images were then partitioned in the ratio of 9:1 to create the corresponding training and testing corpora.

Generation of the DVS spikes responses along with the filtering was implemented entirely in Matlab, while the SCNN and its various layers were coded in Python using the Nengo-DL library [11]. The generated .mat files of the dataset were loaded into the Python network using the Scipy library [22] and the experiments were carried out on the GPU accessed via Google Colaboratory [23].

A. Quantitative Effects of Filtering

To assess the effects of incorporating the neural filtering on DVS recordings, we ran two experiments with the SCNN for a total duration of 60 timesteps. The empirical results are summarized in Table II. For the first scenario of using unfiltered DVS frames, the model achieves an accuracy of 65% which is significantly lower than the values in the remaining rows that correspond to the filtered DVS inputs. This demonstrates that introducing the foveal-pit inspired neural filtering into our retinal model leads to a considerable improvement of 35% even for simplistic visual patterns such as those in our dataset.

TABLE II
Accuracies (%) for the frame-based DVS input

Scenario	Cell - Type	CircShift	Accuracy
Unfiltered	-	-	65.0 %
Filtered	off-center midget	0	77.5 %
	on-center midget		85.0 %
	off-center parasol		92.5 %
	on-center parasol		87.5 %
Filtered	off-center midget	1	77.5 %
	on-center midget		85.0 %
	off-center parasol		100.0 %
	on-center parasol		85.0 %

Amongst the filtered outputs, the parasol ganglion cells have a comparatively higher increase in accuracy compared to the midget cells. Since the parasol cells have larger dimensions and capture the overall background information, they lead to a significant improvement in classification of distinct patterns without intricate details. Thus, the parasol cells lead to a larger performance gain achieving a highest accuracy in the shifted case, as our dataset is composed of only vertical black and white bars. On the other hand, midget cells have smaller dimensions which allows them to capture only the finer details of an image and hence they contribute lesser to the overall increase in classification accuracy.

From Table II, we can also observe that the variations in accuracy for filtering with different ganglion cell types is almost comparable for both the cases of with and without any circular-shift; circular-shift refers to the case where the DoG filters are circular shifted to ‘wrap’ on the raster at all four edges. This reduces artifacts due to edges. The alternative scenario is zero padding at all four edges. The circular-shift value in the Table II is set to 1 indicating all cases where filtering was performed using circular shift at edges, and is set to 0 otherwise.

B. Qualitative Effects of Filtering

For analysing the qualitative effects of neural filtering, we generated raster plots using the analog vector representation of DVS responses as shown in Fig. 4. The neuron numbers range from a value of 0 to 16384 as they represent the pixels of the 128×128 sized electronic retina used in our experiments. The blue markers depict a positive event corresponding to an increase in illumination as the moving edges go from black to white. On the contrary, the red markers represent a negative event and indicate a decrease in illumination as the edges go from white to black.

Figure 4(a) illustrates the unfiltered scenario which has the least distinction between positive and negative events of the input stimulus. This lack of differentiation between black and white bars of the visual pattern also leads to a drop in the classification accuracy which was previously observed in Table II. Further, in the filtered raster plots of Fig. 4, we note that all the ganglion cell filters capture edges more effectively compared to the unfiltered case as there is a clear distinction in the positive and negative events of the filtered raster plots. This improved distinction also leads to a higher classification performance as seen in Section III-A.

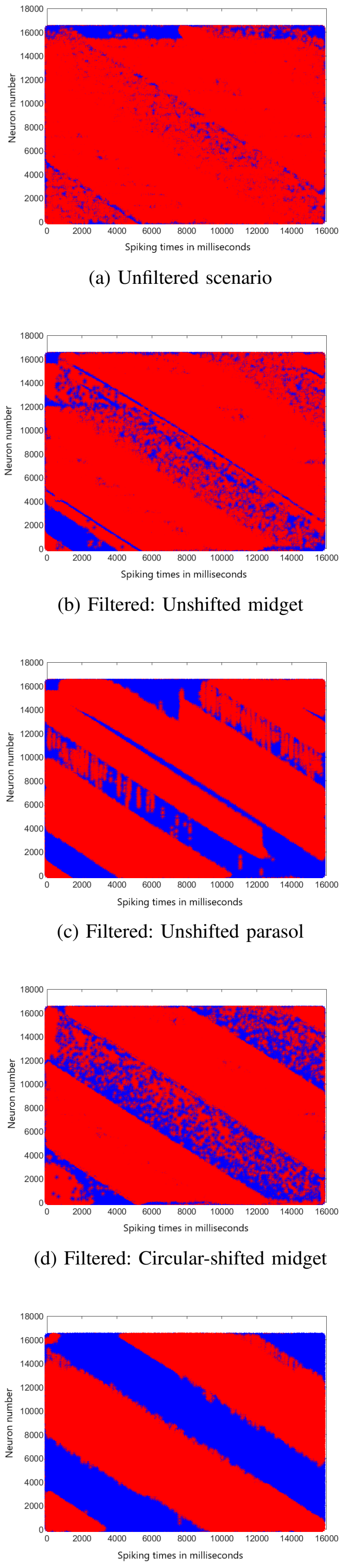
Amongst the outputs of the filtered DVS responses, Fig. 4(b) and Fig. 4(c) represent the cases having a circular-shift value of 0 while Fig. 4(d) and Fig. 4(e) correspond to a circular-shift value of 1. The variation in filtered responses are almost similar for these two cases as seen in Table II. However, the result obtained using the unshifted parasol cell in Fig. 4(c) has lesser clarity between the positive and negative events and hence contains less distinguishable edges than in Fig. 4(e) using the circular-shifted filter. Thus, the best classification accuracy of 100% in Section III-A was achieved using the circular-shifted parasol filter as it captures the changes in illumination more effectively.

Additionally, we observe that the plots generated using midget ganglion cells leads to a noisier qualitative output than those using the parasol ganglion cells. This is because the midget cells have smaller dimensions and are able to pick up only the finer details of an image. But since the simplistic visual patterns used in our experiments lack any intricate details, the midget cells contribute lesser to the overall improvement in the classification performance. Thus, the highest accuracies in Table II are obtained using the parasol cells as they capture the larger and more significant information contained in the input data.

IV. Conclusion and Future Work

In this paper, we have presented a novel method for processing the DVS spike responses of a visual pattern with foveal-pit inspired DoG filters that simulate the primate retinal system. The pattern was composed of varying number of vertical white and black bars of different spatial frequencies moving at a fixed velocity. The outputs from the sensor are applied as input to the bio-inspired neural filters that model the receptive field structure of midget and parasol ganglion cells of the foveal-pit. These processed features are passed as input to our spiking convolutional neural network architecture which classifies the frame-based version of the filtered responses into seven corresponding categories. The SCNN is composed of convolutional and pooling layers and is trained with a modified backpropogation algorithm using a differentiable approximation of spiking neurons [10].

The proposed model demonstrates the effect of applying neural filtering to real DVS data generated from a neuromorphic vision sensor. This builds upon our previous work [15] that depicted the results of foveal-pit inspired filtering for synthetically generated datasets like MNIST [3] and Caltech [16]. Our model achieves a promising performance of 92.5% using the unshifted off-center parasol ganglion cell and an accuracy of 100% in the circular-shifted scenario, which is an improvement of 35% over the classification using unfiltered DVS responses. The empirical results indicate the importance of the foveal-pit inspired neural filtering in redundancy reduction of the DVS inputs and in discarding irrelevant background information.



(a) Unfiltered scenario

(b) Filtered: Unshifted midget

(c) Filtered: Unshifted parasol

(d) Filtered: Circular-shifted midget

(e) Filtered: Circular-shifted parasol

Fig. 4. Raster plot of the (a) unfiltered DVS spike response and filtered DVS response using the off-center (b) unshifted midget cell (c) unshifted parasol cell (d) circular-shifted midget cell (e) circular-shifted parasol cell.

For our proposed network, the asynchronous DVS recordings generated from the first stage of the model were converted to an analog vector representation for training the frame-based classifier composed of convolution layers. As future work, we plan to adapt our spiking convolutional network architecture to directly process event-based data and evaluate the effects of the bio-inspired neural filtering on continuous outputs of a neuromorphic DVS. Also, the dataset used in this work is limited in terms of variation in the inputs as well as the size of the training and testing corpora. Hence, we would like to further verify the effect of the DoG filters on DVS spike responses of larger and more complex datasets.

V. Acknowledgements

This work was funded by the EU grant PCI2019-111826-2 "APROVIS3D", by Spanish grant from the Ministry of Science and Innovation PID2019-105556GB-C31 "NANOMIND" (with support from the European Regional Development Fund) and by the CSIC 2018-50E008 AVE project. BSB is supported by the Science and Engineering Research Board (SERB) of India Fund CRG/2019/003534.

References

[1] Y. Bengio, Y. LeCun *et al.*, "Scaling learning algorithms towards ai," *Large-scale kernel machines*, vol. 34, no. 5, pp. 1–41, 2007.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[3] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[5] A. R. Young, M. E. Dean, J. S. Plank, and G. S. Rose, "A review of spiking neuromorphic hardware communication systems," *IEEE Access*, vol. 7, pp. 135 606–135 620, 2019.

[6] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "Stdp-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56–67, 2018.

[7] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International joint conference on neural networks (IJCNN)*. ieee, 2015, pp. 1–8.

[8] E. O. Neftci, C. Augustine, S. Paul, and G. Detorakis, "Event-driven random back-propagation: Enabling neuromorphic deep learning machines," *Frontiers in neuroscience*, vol. 11, p. 324, 2017.

[14] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in neuroscience*, vol. 9, p. 437, 2015.

[9] H. Mostafa, "Supervised learning based on temporal coding in spiking neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 7, pp. 3227–3235, 2017.

[10] E. Hunsberger and C. Eliasmith, "Training spiking deep networks for neuromorphic hardware," *arXiv preprint arXiv:1611.05141*, 2016.

[11] T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. C. Stewart, D. Rasmussen, X. Choo, A. Voelker, and C. Eliasmith, "Nengo: a python tool for building large-scale functional brain models," *Frontiers in neuroinformatics*, vol. 7, p. 48, 2014.

[12] L. Camunas-Mesa, C. Zamarreno-Ramos, A. Linares-Barranco, A. J. Acosta-Jimenez, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven multi-kernel convolution processor module for event-driven vision sensors," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 2, pp. 504–517, 2011.

[13] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128×128 1.5% contrast sensitivity 0.9% fpn 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, 2013.

[15] S. T. Gupta and B. S. Bhattacharya, "Implementing a foveal-pit inspired filter in a spiking convolutional neural network: a preliminary study," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[16] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *2004 conference on computer vision and pattern recognition workshop*. IEEE, 2004, pp. 178–178.

[17] R. V. Rullen and S. J. Thorpe, "Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex," *Neural computation*, vol. 13, no. 6, pp. 1255–1283, 2001.

[18] B. S. Bhattacharya and S. B. Furber, "Biologically inspired means for rank-order encoding images: A quantitative analysis," *IEEE transactions on neural networks*, vol. 21, no. 7, pp. 1087–1099, 2010.

[19] B. S. Bhattacharya, T. Serrano-Gotarredona, L. Balassa, A. Bhattacharya, A. B. Stokes, A. Rowley, I. Sugiarto, and S. Furber, "A spiking neural network model of the lateral geniculate nucleus on the spinnaker machine," *Frontiers in neuroscience*, vol. 11, p. 454, 2017.

[20] "Jaer project," 2007. [Online]. Available: http://jaerproject.net/

[21] E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data," *Frontiers in neuroscience*, vol. 11, p. 350, 2017.

[22] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[23] "Colaboratory." [Online]. Available: https://colab.research.google.com/