The Journal of Open Source Software

Pyrception: A Python package for biologically plausible neuromorphic perception

Alexander Hadjiivanov [1,2] and **Giulia D'Angelo** [3]

1 Advanced Concepts Team, European Space Agency **2** Adapsent **3** Czech Technical University in Prague

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 09 April 2025
Published: unpublished

Summary

Neuromorphic perception is enjoying a resurgence in attention from the scientific community as event-based cameras (ECs) (Lichtsteiner et al., 2008) become increasingly capable and commercially available (Posch et al., 2014). ECs build upon decades of research on retinomorphic vision, display superior characteristics compared to frame-based cameras (FCs) and highlight the advantages of neuromorphic perception. Here, we present `Pyrception` – a Python library for bio-plausible retinomorphic modelling, simulation and processing of raw visual input. The library builds upon a generalisation of sparse convolution using Toeplitz matrices (outlined below), which allows for efficient processing and modelling of various features of the mammalian retina, such as complex receptive fields, eccentricity-dependent receptive field sizes and log-polar cell arrangement (Chessa et al., 2016).

Statement of need

Hardware prototyping is a difficult and time-consuming process. This is especially valid for neuromorphic vision sensors as the technology is new, and investing effort into developing a new sensor implementing innovative on-sensor features may not be justified unless it has already been demonstrated that the processing pipeline works as intended. `Pyrception` can be used to simulate retinal processing circuits, with envisioned applications in the fields of neuromorphic vision and neuroscience. The ability to perform accurate simulations of various retinal circuits is an efficient way to demonstrate any potential advantages of such circuits and subsequently presenting a strong case for their implementation in hardware. Similarly, such simulation can prove beneficial in neuroscience, for instance, for modelling atypical or poorly understood visual mechanisms to gain insight into the causes and mechanisms behind of certain vision-related medical conditions.

Implementation

The implemention of convolution in deep learning libraries such as PyTorch (Paszke et al., 2017) and TensorFlow (Abadi et al., 2015) is based on the `Im2Col` mechanism (Chellapilla et al., 2006). Since in the case of digital visual input the convolution operation is discrete, in the `Im2Col` approach the input (an image) is partitioned into patches corresponding to each convolved location, which is determined by factors such as kernel size, stride, dilation and so forth. Each of these patches is unrolled into a column vector and concatenated to form the columns of a dense matrix, hence the term `Im2Col` (Fig. 1a). The same operation is performed on all kernels – they are unrolled and concatenated as the *rows* of a separate dense matrix (Fig. 1b). This operation reduces the convolution operation to a simple dense matrix-matrix multiplication.

While the `Im2Col` approach makes the convolution operation well suited for implementation on GPUs, it has several limitations that make it unsuitable for modelling the working principle of the retina. For instance, `Im2Col` requires that all kernels be of the same size, precluding the emulation of log-polar dependent receptive field architecture. Similarly, it makes it exceedingly difficult to implement non-rectangular (e.g., elliptic or irregular) kernels.

In contrast, in `Pyrception` the image is not segmented into patches. Instead, it is unrolled into a single long vector by concatenating all columns of the image (Fig. 1c). As a consequence, the parts of the image that correspond to the receptive field of a kernel are no longer contiguous. To be able to still perform convolution as a simple matrix-matrix (strictly, matrix-vector) multiplication, each corresponding kernel must be unrolled into a *sparse* row in such a way that multiplying the sparse row with the entire image column vector gives the same result as if performed with the `Im2Col` approach (Fig. 1d). This type of convolution is based on Toeplitz matrices (Gnacik & Łapa, 2022), where all diagonals contain identical elements, which is equivalent to shifting consecutive rows by one element to the right. In `Pyrception`, kernels are unravelled into sparse rows, one for each convolved patch of the image, resulting in a sparse Toeplitz matrix.

The versatility of `Pyrception` stems from using a *generalised* version of the sparse convolution approach with a Toeplitz matrix. The 'generalised' part refers to the fact that each patch of the image can be convolved with a *different* kernel since each sparse rows in the kernel matrix encodes an entire kernel. Since each row in the kernel matrix has the same length as the image vector, each kernel can potentially have a full 'view' of the entire image. This gives us complete freedom to choose the parameters of each kernel independently, making it possible to construct convolutional matrices populated with arbitrary kernels with different sizes, orientations and arrangements. Notably, the kernels themselves can be sparse, making it possible to convolve sparse inputs (e.g., inputs arriving from an event camera).

Currently, `Pyrception` implements all five major layers of the mammalian retina, each of which can be emulated with different types of cell arrangements and dynamics, as well as excitatory / inhibitory connections forming receptive fields of different shapes, sizes and orientations. With the default parameters, the `Pyrception` layers implement a generic retinal processing pipeline consisting of a receptor layer followed by horizontal, bipolar, amacrine and ganglion cell layers (Masland, 2011). By simply deriving from the existing layer classes and overriding the `forward()` method, it is possible to implement different retinal circuits, such as motion segmentation (Baccus et al., 2008; Clerico et al., 2024; Ölveczky et al., 2003), looming detection for detecting approaching objects (Gollisch & Meister, 2010), optical flow estimation and visual tracking (Angelo et al., 2025; Javier Traver & Bernardino, 2010). We welcome contributions to `Pyrception` that would make the library as useful as possible, including implementing bio-plausible modalities other than vision.

The Journal of Open Source Software

Figures

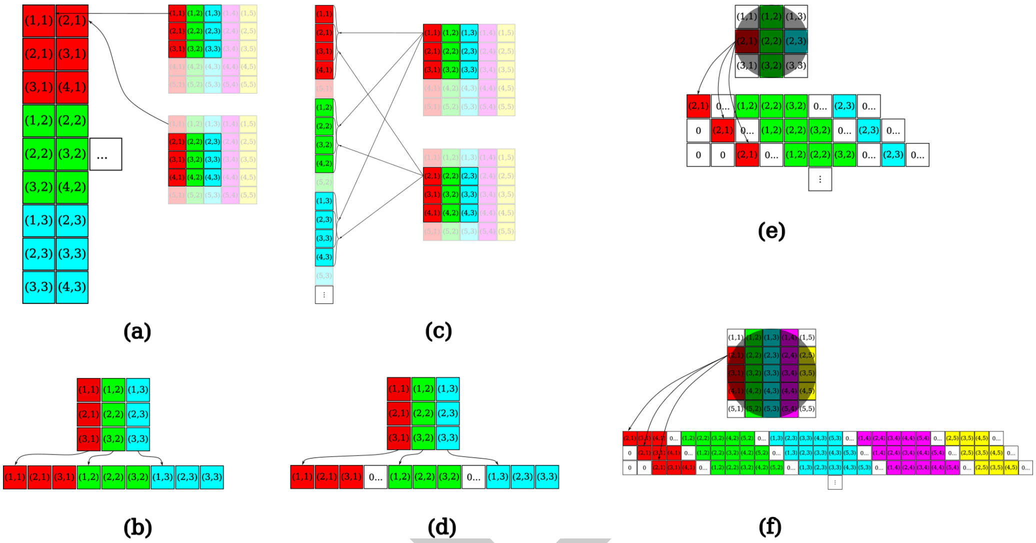

Figure 1: Visualisation of dense and sparse operations on images and kernels for convolution. (a) Unrolling image patches into a dense matrix (`Im2Col` operation) and (b) a corresponding kernel into a dense row (`Ker2Row` operation). (c) Unrolling an image column-wise into a vector (`Im2Vec` operation). The (non-contiguous) elements corresponding to the same patches as those in (a) are illustrated. (d) Unrolling a kernel into a sparse row (`Ker2SpRow` operation). (e, f) Examples of circular kernels with radii of 3 and 5 pixels, respectively, with the corresponding unrolled `Ker2SpRow` representations shown below each kernel.

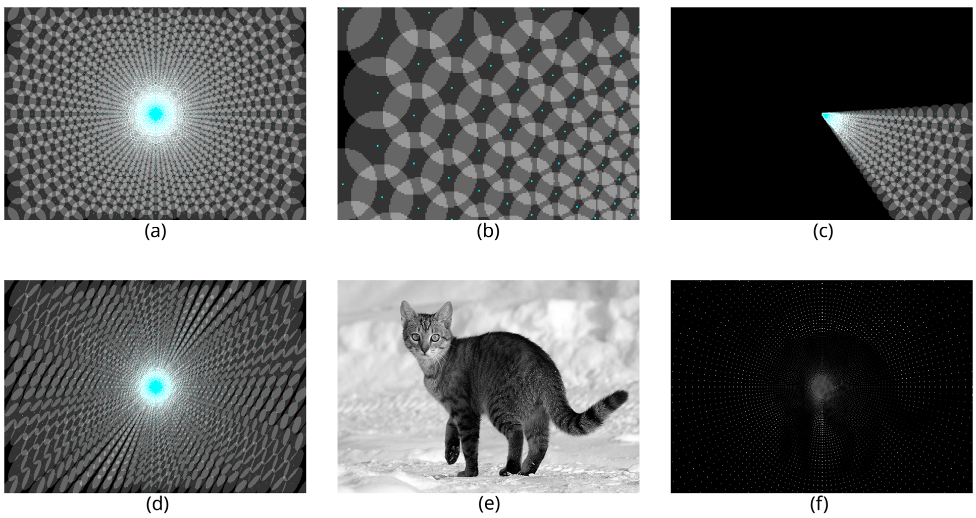

Figure 2: A visualisation of kernels of different sizes and orientations. (a) Kernels arranged and a phyllotactic log-polar distribution (similar to the seeds of a sunflower). Kernels towards the center of the image are smaller than ones towards the edges, emulating the eccentricity-dependent receptive field size observed in the mammalian retina. Due to the different sizes of the kernels, the 'stride' (and therefore the overlap) is non-uniform. (b) A zoomed-in version of the top left corner of the plot in (a). (c) A plot of the kernels for a subset of 10 sectors of kernels (a sector is a group of kernels whose centers are located along the same radial line). (d) The same kernel arrangement as in (a), but with elliptic kernels oriented at 45°. (e) A raw image (credit: Von.grzanka / Wikimedia) and (f) the corresponding activation map resulting from convolving the image with the kernel map in (a).

Acknowledgements

A.H. is grateful for the support of the Advanced Concepts Team of the European Space Agency, where the conceptualisation and a large portion of the development of this library took place. G.D. acknowledges the financial support from the European Union's HORIZON-MSCA-2023-PF-01-01 research and innovation programme under the Marie Skłodowska-Curie grant agreement ENDEAVOR No 101149664.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*. https://www.tensorflow.org/

Angelo, G. D., Clerico, V., Bartolozzi, C., Hoffmann, M., Furlong, P. M., & Hadjiivanov, A. (2025). Wandering around: A bioinspired approach to visual attention through object motion sensitivity. *arXiv Preprint arXiv:2502.06747*.

Baccus, S. A., Ölveczky, B. P., Manu, M., & Meister, M. (2008). A retinal circuit that computes object motion. *Journal of Neuroscience*, *28*(27), 6807–6817.

Chellapilla, K., Puri, S., & Simard, P. (2006). High Performance Convolutional Neural Networks for Document Processing. In G. Lorette (Ed.), *Tenth International Workshop on Frontiers in Handwriting Recognition*. Suvisoft. https://inria.hal.science/inria-00112631

Chessa, M., Maiello, G., Bex, P. J., & Solari, F. (2016). A space-variant model for motion interpretation across the visual field. *Journal of Vision*, *16*(2), 12–12.

Clerico, V., Snyder, S., Lohia, A., Kaiser, M. A.-A., Schwartz, G., Jaiswal, A., & Parsa, M. (2024). *Retina-inspired object motion segmentation for event-cameras*. https://arxiv.org/abs/2408.09454

Gnacik, M., & Łapa, K. (2022, October 27). *Using Toeplitz Matrices to obtain 2D convolution*. https://doi.org/10.21203/rs.3.rs-2195496/v1

Gollisch, T., & Meister, M. (2010). Eye Smarter than Scientists Believed: Neural Computations in Circuits of the Retina. *Neuron*, *65*(2), 150–164. https://doi.org/10.1016/j.neuron.2009.12.009

Javier Traver, V., & Bernardino, A. (2010). A review of log-polar imaging for visual perception in robotics. *Robotics and Autonomous Systems*, *58*(4), 378–398. https://doi.org/10.1016/j.robot.2009.10.002

Lichtsteiner, P., Posch, C., & Delbruck, T. (2008). A 128X128 120 dB 15us latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, *43*(2), 566–576.

Masland, R. H. (2011). Cell Populations of the Retina: The Proctor Lecture. *Investigative Opthalmology & Visual Science*, *52*(7), 4581–4591. https://doi.org/10.1167/iovs.10-7083

Ölveczky, B. P., Baccus, S. A., & Meister, M. (2003). Segregation of object and background motion in the retina. *Nature*, *423*(6938), 401–408.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). *Automatic differentiation in PyTorch*.

Posch, C., Serrano-Gotarredona, T., Linares-Barranco, B., & Delbruck, T. (2014). Retinomorphic Event-Based Vision Sensors: Bioinspired Cameras With Spiking Output. *Proceedings of the IEEE*, *102*(10), 1470–1484. https://doi.org/10.1109/JPROC.2014.2346153