Sparse Vector Binding on Spiking Neuromorphic Hardware Using Synaptic Delays

Alpha Renner
alpren@ini.uzh.ch
Institute of Neuroinformatics,
University of Zurich and ETH Zurich
Switzerland

Yulia Sandamirskaya
yulia.sandamirskaya@intel.com
Intel Neuromorphic Computing Lab
Zurich, Switzerland

Friedrich T. Sommer
fsommer@berkeley.edu
Intel Neuromorphic Computing Lab
Redwood Center for Theoretical Neuroscience,
UC Berkeley
USA

E. Paxon Frady
e.paxon.frady@intel.com
Intel Neuromorphic Computing Lab
USA

ABSTRACT

Vector Symbolic Architectures (VSA) were first proposed as connectionist models for symbolic reasoning, leveraging parallel and in-memory computing in brains and neuromorphic hardware that enable low-power, low-latency applications. Symbols are defined in VSAs as points/vectors in a high-dimensional neural state-space. For spiking neuromorphic hardware (and brains) particularly sparse representations are of interest, as they minimize the number of costly spikes. Furthermore, sparse representations can be efficiently stored in simple Hebbian auto-associative memories, which provide error correction in VSAs. However, the binding of spatially sparse representations is computationally expensive because it is not local to corresponding pairs of neurons as in VSAs with dense vectors. Here, we present the first implementation of a sparse VSA on spiking neuromorphic hardware, specifically Intel's neuromorphic research chip Loihi. To reduce the cost of binding, a delay line and coincidence detection are used, trading off space with time. We show as proof of principle that our network on Loihi can perform the binding operation of a classical analogical reasoning task and discuss the cost of different sparse binding operations. The proposed binding mechanism can be used as a building block for VSA-based architectures on neuromorphic hardware.

CCS CONCEPTS

• **Hardware** → **Neural systems**.

KEYWORDS

Spiking Neural Networks, Vector Symbolic Architecture (VSA), Hyperdimensional Computing, Neuromorphic Hardware, Binding, Coincidence detection, Sparse distributed code

ICONS 2022, July 27–29, 2022, Knoxville, TN, USA

ACM ISBN 978-1-4503-9789-6/22/07…$15.00
https://doi.org/10.1145/3546790.3546820

ACM Reference Format:
Alpha Renner, Yulia Sandamirskaya, Friedrich T. Sommer, and E. Paxon Frady. 2022. Sparse Vector Binding on Spiking Neuromorphic Hardware Using Synaptic Delays. In *International Conference on Neuromorphic Systems (ICONS 2022), July 27–29, 2022, Knoxville, TN, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3546790.3546820

1 INTRODUCTION

Vector Symbolic Architectures (VSA) [8, 11, 19], also known as hyperdimensional computing [12], have been proposed as a computing framework for emerging hardware such as spiking neuromorphic hardware [4, 14].

VSAs allow a connectionist (neural network) implementation of symbolic computation. Therefore, they make symbolic computation compatible with the parallel and in-memory computing that comes with the large number of simple computing elements (neurons), and that makes neuromorphic hardware intriguing for low-power, low-latency applications. In VSAs, symbols and functions are defined as vectors in a high dimensional state-space. Each element of the vectors can be seen as a neuron and information is distributed throughout the population of neurons. The high dimensionality of vectors in VSA makes randomly chosen vectors almost orthogonal, in other words they are dissimilar to each other.

VSAs define two main operations on pairs of these vectors which keep the vector length and dimensionality constant: "Bundling" ($\oplus$) allows the representation of sets of symbols by creating a new vector that is similar to all elements of the set. "Binding" ($\otimes$) takes two vectors and generates a new unique vector that is dissimilar to each original component. This new vector represents the conjunction of the two original symbols. Binding can be regarded as a compressed outer product [6, 28], where every possible combination of original values has its own unique representation (of fixed length and dimensionality). These operations are used to associate different symbols (e.g. filler with a role or object with a location) to create data structures (variables, trees, lattices) and to solve cognitive reasoning tasks (such as analogies).

For neuromorphic hardware, sparse representations (few non-zero elements) are of interest, as they minimize the cost associated with spikes. Furthermore, sparse representations can represent

larger sets of symbols and show increased capacity for learning [7, 10, 25]. Presumably for similar reasons, biological neural activity is often sparse [17].

However, spatially sparse representations in VSA require a binding operation that is not local to corresponding pairs of neurons, which is more expensive in terms of synaptic resources compared to dense VSA binding operations. This issue we address on neuromorphic hardware in this work. In particular, we examine binary sparse block codes [6, 15, 21] that have been proposed as a sparse VSA. Block codes could be regarded as a compromise between distributed and sparse localist representations, combining the best of both worlds. The binary block code VSA has been shown to be one of the most efficient VSAs for representing and computing on large sets (bundling) [26], and it is also cost efficient and easy to implement on digital hardware due to its binary encoding [15]. The binding operation for block codes is local circular convolution (LCC) [6], which corresponds to modulo addition of the active neuron indices within blocks – e.g. if neuron 0 and neuron 2 are active in the first vector and neuron 2 in the second vector, then in the bound vector, neuron $(0+2) \bmod 5 = 2$ and neuron $(2+2) \bmod 5 = 4$ are active (see Fig. 1).

Here, we present the first implementation of a sparse VSA binding operation on spiking neuromorphic hardware, specifically Intel's neuromorphic research chip Loihi [2]. To reduce the memory cost of binding, we propose a model that trades off space against time, where delay lines and coincidence detection are used to compute the binding operation. We also show, as proof of principle, that our network on Loihi can perform the binding operation of a classical analogical reasoning task and discuss the cost of different sparse binding operations.

2 MODEL AND RESULTS

A vector of a binary block code [6, 15, 21] has N neurons and is segmented into K blocks of equal length $L = N/K$. In its sparsest form, each block has a single non-zero element (corresponding to a spike). The binding operation for the block code is local (block-local) circular convolution (LCC). It is denoted by $\otimes$, and $C = A \otimes B$ is defined so that activations (spikes) in each block of vector A are circularly shifted by the active indices in B [15]. Assuming a single activation in B, this is equivalent to the modulo sum of the indices of the two operands.

The neural implementation of LCC usually requires an "outer product" circuit for each block as illustrated in Fig. 1A (see [29] for a Loihi implementation of a single such circuit). The two input layers feed into a middle layer of simple coincidence neurons along the diagonal and vertical directions, and the middle layer activity is then summed in the horizontal direction. Neurons in the middle layer only reach the firing threshold (black) if they receive coincident input from both input layers.

For each block, this outer product layer requires L^2 neurons and $3L^2$ synapses. The implementation proposed here, as illustrated in Fig. 1B, uses time instead of space in the middle layer to realize the outer product. For this, each neuron in the input layers is connected to the middle layer (of length L instead of L^2) with a delay line replicating the pattern shown in Fig. 1B. The neurons in the middle layer that receive coincident inputs from the two input layers fire and send their spikes to the output layer, which integrates its input over time. At the end of an iteration, The output layer gets "queried" by an additional sub-threshold input. This query input triggers all neurons in the output layer that have received inputs from the middle layer during the last iteration to fire.

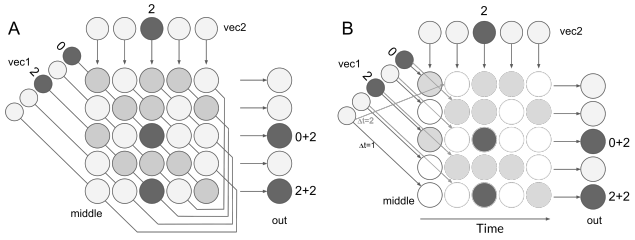

Figure 1: Schematic that visualizes the mechanisms of circular convolution (calculating the sum of the indices). Here, the sum of (0,2) and (2) is calculated, resulting in an output of (2,4). Neurons that fire are shown in black, neurons that only receive sub-threshold input in gray, and neurons that receive no input at all in white. A. Full outer product in space that needs N^2 neurons but allows immediate readout of the result in the horizontal direction. The diagonal lines that pass behind neurons signify separate connections from the source to each connected neuron in the middle layer. B. Outer product in time that only needs N neurons in the middle layer. The connections from the input layers to the middle layer are delayed in order to produce the same pattern as in A. To avoid clutter, only the synapses with a delay of 1 and 2 are shown between the first input vector and the middle layer.

This circuit could be implemented as described, however, we added some additional improvements to save connectivity and timesteps in the on-chip implementation, which we show in Fig. 2A. Each block is first encoded into a single neuron ('inter') by transforming activity in space (i.e., which neuron in the layer is active) into activity in a time window. The first neuron is connected with a delay of 1 timestep, the second with a delay of 2 timesteps, and so on. Furthermore, we split the middle layer into two parts with opposite delays, one analogous to the neurons below the diagonal in Fig. 1 and the other one on and above the diagonal. The full connectivity of the module is shown in Fig. 2A and the spikes that are generated during the binding process are shown in a raster plot in Fig. 2B.

The neuron model on Loihi is a current-based integrate-and-fire model (see Eq. 3 and 4). Apart from the output neurons, which integrate inputs from the middle layer over the whole cycle, the current and voltage decay time constants were set to 1 timestep, such that there is no memory, and coincidences are detected in the same timestep (see Methods section for details).

To demonstrate the functionality of our binding module implemented on the Loihi chip, we performed binding of a larger VSA vector of $K = 80$ blocks, and $L = 20$ neurons per block. We implemented the example from the well-known tutorial "What's the Dollar of Mexico?" [13], which demonstrates how analogical reasoning is performed with VSA. Here, data vectors of role-filler pairs are created representing concepts for different countries, e.g. $USTATES =$

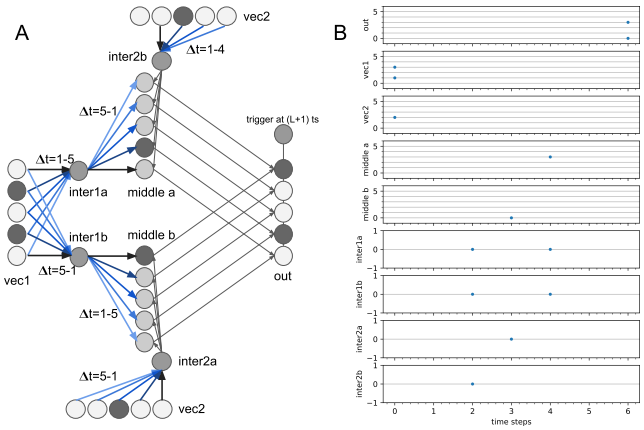

Figure 2: A. One block of the optimized LCC binding module as implemented on Loihi. Synapses without delay ($\Delta t = 1$) are shown in black and delayed synapses are shown in shades of blue (lighter blue means longer delay). Note that $\Delta t = 5 - 1$ labels a group of synapses with delays ranging from 5 to 1 timesteps (with a decrement of 1). B. Raster plot of the neurons of the module shown in A. In timestep 0, the two input layers fire, and in the last timestep, the output layer is triggered and fires according to the modulo L addition of the input indices: $(3 + 2) \bmod 5 = 0$ and $(1 + 2) \bmod 5 = 3$.

$CAP \otimes DC + CUR \otimes DOL$, $MEX = CAP \otimes MXC + CUR \otimes PES$. In this example, the role vectors CAP and CUR indicate the capital and currency, which are attached to the particular filler variables for each country, i.e. $CUR \otimes DOL$ indicates the currency is dollar. A transformation vector, $F_{UM} = MEX \otimes USTATES^{-1}$ can be created that forms an analogical mapping between the data vectors. To answer the question "What's the Dollar of Mexico?", the vector that represents "dollar" (DOL) is used to query the vector that represents the mapping from US to Mexico (F_{UM}). The output can then be decoded with the expected result of "peso" (PES):

$$DOL \otimes F_{UM} = PES + noise \approx PES \tag{1}$$

$$\tag{2}$$

Fig. 3A illustrates the network for implementing the analogical reasoning task. After the mapping vector is queried, the output is compared to a codebook of vectors representing the relevant concepts. The highest inner-product between the output and the vectors of the codebook produces the answer.

3 DISCUSSION

The proposed implementation of LCC binding trades off space with time on the algorithm level, saving chip area as the size of the required neural network scales linearly instead of quadratically as the outer product layer. Instead, the binding takes more time - the response time scales with the size of the blocks and the timestep duration on Loihi. Since the timestep can be in the low microseconds range, the binding is performed in less than a millisecond with coincidence windows of a single timestep. For a comparison of different methods for implementing sparse binding, including an example calculation, see Table 1.

Table 1: Comparison of binding methods. For "blocks in time", timesteps scale with L avoiding quadratic scaling of the number of neurons and synapses.

	full outer product	blocks in space	blocks in time
neurons	$I^2 = 100 \cdot 10^6$	$L^2K = 10^6$	$6LK = 60 \cdot 10^3$
synapses	$3I^2 = 300 \cdot 10^6$	$3L^2K = 3 \cdot 10^6$	$10LK = 100 \cdot 10^3$
timesteps	3	3	$L + 3 = 103$
spikes	$3S = 30$	$3SK + K = 3100$	$5SK + 3K = 5300$

I = 10000 : Total number of symbols, L = 100: Neurons per block,
K = 100 : Number of blocks, S = 10 : Bundled symbols
Encoding to block code: $K \cdot I = 10^6$ synapses

Note that the approach described here is similar to another VSA, Fourier Holographic Representations (FHRR). FHRR does not used blocks, but dense complex vectors, however from the perspective of binding, a single complex number (phasor) of FHRR is comparable to a block of a block code. The difference is, however, that in block codes several active elements (spikes) per block can occur, while in FHRR (as bundling corresponds to complex addition instead of a logical AND) only a single spike per neuron is allowed (per cycle) which simplifies the binding operation significantly. We implemented this alternative VSA and binding operation on Loihi with spike timing-based phasor neurons in [22]). The advantage of the block code comes in the energy efficiency of sparse representations (spikes being transmitted), as well as fewer memory requirements for the sparse associative memories.

The benefit of sparse block code binding is that it can save orders of magnitude of neurons and synapses compared to a full outer product binding, but it is more resource expensive than dense Hadamard binding. However, the sparsity has benefits in reducing the number of spikes being transmitted, and lowering the number of synaptic connections in the codebooks and clean-up memories of VSA algorithms.

From an engineering perspective, it is worth noting that delays on neuromorphic hardware can be expensive. In digital hardware, a shift register has to be reserved to store incoming spikes of each neuron, limiting the maximally available delay [2] and therefore the possible block length with the proposed binding mechanism. This is in contrast to biological systems that can make use of the difference in signal conduction speeds of dendrites and axons with different properties (e.g., diameter) to realize a delay line. The kind of coincidence detection we use here for binding is seen in a broader context in neuroscience, e.g. delay and coincidence detection of input from the two ears [1, 9] for sound localization (neuromorphic implementation [16]) and for combining the inputs of the eyes for stereo vision [18, 24].

In conclusion, the proposed binding module could serve as a building block to enable the implementation of larger VSA architectures (such as [5]) on neuromorphic hardware with a limited amount of neurons and synapses. Models that use the full outer product representation e.g. for coordinate transforms [3, 20, 27] could be redesigned into this framework and benefit from the proposed mechanism. In recent work, we have demonstrated applications of VSA in spatial processing [22] and visual SLAM [23] using

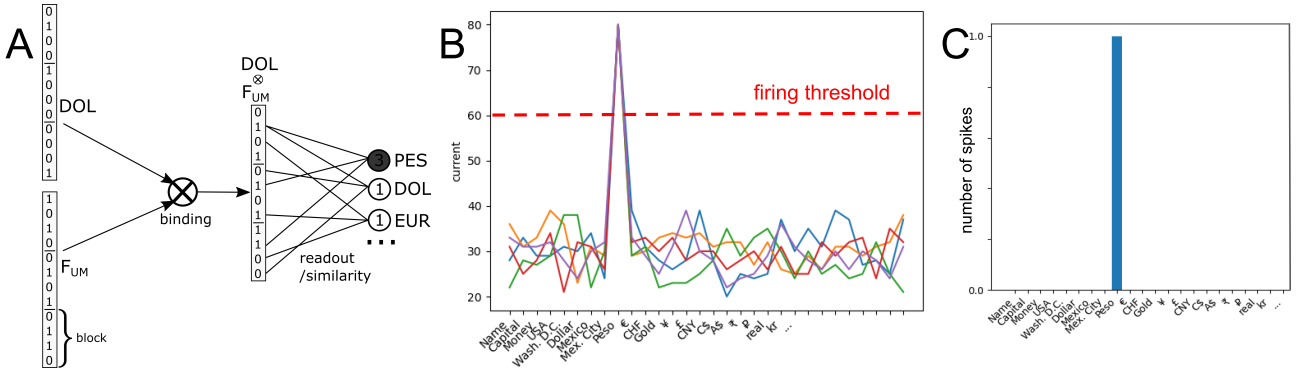

Figure 3: A. Schematic of the network that was implemented on the Loihi chip. The two block-vectors DOL and F_{UM} are bound by K=80 instances of the binding module shown in Fig. 2. The resulting vector is then read out into a layer that corresponds to all possible symbols. Note that DOL is a sparse vector with a single active neuron per block, while F_{UM} is a bundle with several active neurons per block. B. Input currents to the readout neurons. PES, which is the correct answer, receives the majority of the input. The different colors are repetitions of the same experiment with different randomly generated VSA vectors. Note that to the right of the 9 symbols involved in the task, 51 randomly chosen and arbitrarily labelled symbols are added. C. As only the PES neuron receives input above the firing threshold, it is the only one that fires.

FHRR, which also could be implemented with this framework and would have gains in memory efficiency.

4 METHODS

All experiments on Loihi were run with NxSDK version 0.9.9 on the Kapoho Bay system which contains 2 Loihi chips. Software API and hardware were provided by Intel Labs. Random vectors were created with numpy in python. The current-based (CUBA) integrate and fire model as implemented on Loihi is described by the following discrete-time dynamics:

$$V(t+1) = V(t) - \frac{1}{\tau_V}V(t) + U(t), \tag{3}$$

$$U(t+1) = U(t) - \frac{1}{\tau_U}U(t) + \sum_j w_{ij}\delta_j(t - d_{ij}). \tag{4}$$

V denotes the membrane potential, U, the current, and t, the timestep. The last term in Equation 4 is the input to the current from presynaptic spikes. Each spike arrives after a synaptic delay d_{ij} and increases the current by a synaptic weight w_{ij}.

Delays in the delay lines range from 1 timestep to the block length as shown in Fig. 2. Weights of all synapses are set to 2 apart from the synapses that query the output of the binding module, which are set to 19, just below the firing threshold of the output layer, so that it fires if it has integrated at least one coincidence spike from the binding layer.

A neuron fires when its threshold V_{thr} is reached and is then reset to $V = 0$. We use a threshold of $V_{thr} = 3$ in the binding layer, $V_{thr} = 20$ in the output layer of the binding module, $V_{thr} = 60$ in the readout layer and $V_{thr} = 1$ in all other layers. Both τs are set to 1 timestep in all layers apart from the output layer of the binding module, so that coincidence is only detected in the same timestep. τ_U of the binding output layer is set to 1 ts and τ_V is set to to infinity (no decay). The network shown in the results (see Fig. 3) uses K=80 blocks and L=20 neurons per block.

ACKNOWLEDGMENTS

A.R. was supported by Accenture Labs and the Swiss National Science Foundation (SNSF) grant Ambizione PZOOP2 168183 ELMA. We also thank Intel labs for providing support for and access to the Loihi hardware and software.

REFERENCES

[1] Catherine E Carr and Masakazu Konishi. 1988. Axonal delay lines for time measurement in the owl's brainstem. *Proceedings of the National Academy of Sciences* 85, 21 (1988), 8311–8315.
[2] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Prasad Joshi, Andrew Lines, Andreas Wild, and Hong Wang. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* (2018), 1–1. https://doi.org/10.1109/{MM}.2018.112130359
[3] PU Diehl and M Cook. 2016. Learning and inferring relations in cortical networks. *arXiv preprint arXiv:1608.08267* (2016). https://arxiv.org/abs/1608.08267
[4] Chris Eliasmith and Charles H Anderson. 2003. *Neural engineering: Computation, representation, and dynamics in neurobiological systems.* MIT press.
[5] Chris Eliasmith, Terrence C Stewart, Xuan Choo, Trevor Bekolay, Travis DeWolf, Yichuan Tang, and Daniel Rasmussen. 2012. A large-scale model of the functioning brain. *Science* 338, 6111 (30 nov 2012), 1202–1205. https://doi.org/10.1126/science.1225266
[6] E Paxon Frady, D Kleyko, and FT Sommer. 2020. Variable Binding for Sparse Distributed Representations: Theory and Applications. *arXiv preprint arXiv:2009.06734* (2020). https://arxiv.org/abs/2009.06734
[7] E Paxon Frady and Friedrich T Sommer. 2019. Robust computation with rhythmic spike patterns. *Proceedings of the National Academy of Sciences of the United States of America* 116, 36 (3 sep 2019), 18050–18059. https://doi.org/10.1073/pnas.1902653116
[8] R. W. Gayler. 2003. Vector Symbolic Architectures Answer Jackendoff's Challenges for Cognitive Neuroscience. In *Joint International Conference on Cognitive Science (ICCS/ASCS).* 133–138.
[9] Lloyd A Jeffress. 1948. A place theory of sound localization. *Journal of comparative and physiological psychology* 41, 1 (1948), 35.
[10] Pentti Kanerva. 1988. *Sparse distributed memory.* MIT press.
[11] Pentti Kanerva. 1996. Binary spatter-coding of ordered K-tuples. In *Artificial neural networks —ICANN 96*, Christoph Malsburg, Werner Seelen, Jan C. Vorbrüggen, Bernhard Sendhoff, Gerhard Goos, Juris Hartmanis, and Jan Leeuwen (Eds.). Lecture notes in computer science, Vol. 1112. Springer Berlin Heidelberg, Berlin, Heidelberg, 869–873. https://doi.org/10.1007/3-540-61510-5_146
[12] Pentti Kanerva. 2009. Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive computation* 1, 2 (jun 2009), 139–159. https://doi.org/10.1007/s12559-

009-9009-8

[13] P Kanerva. 2010. What we mean when we say" What's the dollar of mexico?": Prototypes and mapping in concept space. *2010 AAAI fall symposium series* (2010).

[14] D Kleyko, M Davies, EP Frady, and P Kanerva. 2021. Vector Symbolic Architectures as a Computing Framework for Nanoscale Hardware. *arXiv preprint arXiv …* (2021). https://arxiv.org/abs/2106.05268

[15] Mika Laiho, Jussi H. Poikonen, Pentti Kanerva, and Eero Lehtonen. 2015. High-dimensional computing with sparse vectors. In *2015 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 1–4. https://doi.org/10.1109/{BioCAS}.2015.7348414

[16] John Lazzaro and Carver A Mead. 1989. A silicon model of auditory localization. *Neural computation* 1, 1 (1989), 47–57.

[17] Bruno A Olshausen and David J Field. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381, 6583 (1996), 607–609.

[18] Marc Osswald, Sio-Hoi Ieng, Ryad Benosman, and Giacomo Indiveri. 2017. A spiking neural network model of 3D perception for event-based neuromorphic stereo vision systems. *Scientific Reports* 7 (12 jan 2017), 40703. https://doi.org/10.1038/srep40703

[19] TA Plate. 1994. *Distributed representations and nested compositional structure*. University of Toronto, Department of Computer Science. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.5527&rep=rep1&type=pdf

[20] Alexandre Pouget, Sophie Deneve, and Jean-René Duhamel. 2002. A computational perspective on the neural basis of multisensory spatial representations. *Nature Reviews Neuroscience* 3, 9 (2002), 741–747.

[21] D.A. Rachkovskij. 2001. Representation and processing of structures with binary sparse distributed codes. *IEEE transactions on knowledge and data engineering* 13, 2 (2001), 261–276. https://doi.org/10.1109/69.917565

[22] Alpha Renner, Supic Lazar, Andreea Danielescu, Giacomo Indiveri, Bruno A. Olshausen, Yulia Sandamirskaya, Friedrich T. Sommer, and Paxon Frady. 2022. Neuromorphic Visual Scene Understanding with Resonator Networks. *arXiv preprint arXiv* (2022).

[23] Alpha Renner, Supic Lazar, Andreea Danielescu, Giacomo Indiveri, Friedrich T. Sommer, Paxon Frady, and Yulia Sandamirskaya. 2022. Neuromorphic Visual Odometry with Resonator Networks. *arXiv preprint arXiv* (2022).

[24] Nicoletta Risi, Alessandro Aimar, Elisa Donati, Sergio Solinas, and Giacomo Indiveri. 2020. A Spike-Based Neuromorphic Architecture of Stereo Vision. *Frontiers in neurorobotics* 14 (13 nov 2020). https://doi.org/10.3389/fnbot.2020.568283

[25] Edmund T Rolls and Alessandro Treves. 1990. The relative advantages of sparse versus distributed encoding for associative neuronal networks in the brain. *Network: computation in neural systems* 1, 4 (1990), 407.

[26] K Schlegel, P Neubert, and P Protzel. 2020. A comparison of Vector Symbolic Architectures. *arXiv preprint arXiv:2001.11797* (2020). https://arxiv.org/abs/2001.11797

[27] Sebastian Schneegans and Gregor Schöner. 2012. A neural mechanism for coordinate transformation predicts pre-saccadic remapping. *Biological cybernetics* 106, 2 (2012), 89–109.

[28] Paul Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial intelligence* 46, 1-2 (nov 1990), 159–216. https://doi.org/10.1016/0004-3702(90)90007-M

[29] Rasmus Karnoe Stagsted, Antonio Vitale, Alpha Renner, Leon Bonde Larsen, Anders Lyhne Christensen, and Yulia Sandamirskaya. 2020. Event-based PID controller fully realized in neuromorphic hardware: a one DoF study. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 10939–10944. https://doi.org/10.1109/{IROS45743}.2020.9340861