Material and Physical Reservoir Computing for Beyond CMOS Electronics: Quo Vadis?

C. Teuscher*
teuscher@pdx.edu
Portland State University, ECE Department
Portland, OR, USA

ABSTRACT

Traditional computing is based on an engineering approach that imposes logical states and a computational model upon a physical substrate. Physical or material computing, on the other hand, harnesses and exploits the inherent, naturally-occurring properties of a physical substrate to perform a computation. To do so, reservoir computing is often used as a computing paradigm. In this review and position paper, we take stock of where the field currently stands, delineate opportunities and challenges for future research, and outline steps on how to get material reservoir to the next level. The findings are relevant for beyond CMOS and beyond von Neumann architectures, ML, AI, neuromorphic systems, and computing with novel devices and circuits.

CCS CONCEPTS

• **Computer systems organization → Neural networks**.

KEYWORDS

reservoir computing, material computing, neural network, neuromorphic, hardware, beyond CMOS

ACM Reference Format:
C. Teuscher. 2023. Material and Physical Reservoir Computing for Beyond CMOS Electronics: Quo Vadis?. In *18th ACM International Symposium on Nanoscale Architectures (NANOARCH '23), December 18–20, 2023, Dresden, Germany*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3611315.3633251

1 INTRODUCTION

"Information is inevitably tied to a physical representation and therefore to restrictions and possibilities related to the laws of physics and the parts available in the universe" [23]. As a consequence, every computation is physical because the information to be processed needs a physical substrate. The traditional way of building computers is "top-down" and tries to abstract from the physics as much as possible by imposing logical states and a computational model upon the physical substrate. This "designed" way of building computers requires an the ability to control the physical substrate. An alternative way of solving computational problems is to harnesses and exploit the inherent, naturally-occurring properties of a physical substrate to perform a desired computation [6]. This "bottom-up" way of building computers is called *material*, *physical*, *in materia* [41], or *intrinsic* computing.

Before the dawn of digital computers, the analog computers of the early 20^{th} century are perhaps the best example of early *in materia* computing. Such computers would represent data in (quasi) continuous form of physical quantities and perform calculations with analog circuits, such as operational amplifiers. The lack of an explicit memory and central processing unit avoids many of the bottlenecks and inefficiencies of digital von-Neumann-type computers. Because analog computers are highly application-specific, they lack flexibility and are not generally universal. They are also difficult to "program."

A new variant of *in materia* computing emerged in the early nineties in the form of *evolvable hardware*. "Evolution-in-materia (EIM) is a term that refers to the manipulation of physical systems using computer controlled evolution (CCE)" [33]. "It is argued that natural evolution is, par excellence, an algorithm that exploits the physical properties of materials. Such an exploitation of the physical characteristics has already been demonstrated in intrinsic evolution of electronic circuits" [32]. Early pioneers of the field were Yoshihito [52], Thompson [47], Layzell [24], and others. In these early approaches, a desired function was generally attempted to evolve in hardware (in an open-ended, unconstrained, and rather brute-forced way), without employing a specific architecture.

The appearance of the *Reservoir Computing* (RC) paradigm in the early 2000s (independently proposed by two groups) [21, 31] suddenly provided an entirely new way to harness the inherent physical properties of materials. RC has now become an umbrella term for a variety of similar computational models in which a fixed "reservoir," a sub-component of an RC system that shows complex, often non-linear dynamics, and a rich space of internal states, projects the inputs into a high(er) dimensional space. A desired computation is then obtained by a observing a (usually linear, memory-less) projection of the reservoir state onto the low(er) dimensional output space, i.e., the desired output. The "output layer" is the only sub-component of an RC system that is 'trained," the reservoir itself is fixed and left to its own inherent dynamics.

RC has a number of advantages over related approaches: (1) it excels particularly in the area of temporal signal processing and learning dynamical systems; (2) it leads to a significantly lower training complexity compared to traditional recurrent neural networks because only the (linear) output layer is trained; and (3) almost any physical system with interesting-enough dynamics can be used as a reservoir, as long as it has the *fading memory* and the *separation* property: "Another advantage is that the reservoir without adaptive updating is amenable to hardware implementation

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs International 4.0 License.

NANOARCH '23, December 18–20, 2023, Dresden, Germany
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0325-6/23/12.
https://doi.org/10.1145/3611315.3633251

using a variety of physical systems, substrates, and devices. In fact, such *Physical Reservoir Computing* (PRC) has attracted increasing attention in diverse fields of research" [46].

In this review and position paper, we take stock of where the field of physical and material RC currently stands, delineate opportunities and challenges for future research, and outline how we can get there.

2 A MINI REVIEW OF MATERIAL RC SYSTEMS

The fact that reservoirs can be unstructured, imperfect, and unreliable, has drawn increasing attention from the hardware community because RC provides a promising framework to compute with emerging devices and device networks one does not need to have full control over [10, 46]. Defects, faults, variation, and unstructuredness in hardware actually become desired properties because they enhance the dynamic response of a reservoir to its input perturbation [4]. RC has also attracted significant interest in the neuromorphic computing community because of its potential to implement neural networks by directly harnessing the materials, as opposed to building traditional von Neumann architectures that then are engineered to "simulate" neural networks [5, 26].

Perhaps the most exotic reservoir ever proposed is an actual "reservoir:" a bucket of water [12]. Fernando and Sojakka showed in 2003 that such a system can solve both the XOR and the spoken digit recognition task. Since then, many material RC implementations were proposed in several topical areas, including physical reservoir computing with plants [39].

Without claiming this to be a comprehensive review, we shall mention some selected and recent work below in each of these areas. For a comprehensive review of physical RC, see [7, 46].

Mem-element-based RC. Mem-based RC goes back to 2014 [22]. Du et al. [11] experimentally implemented a RC system using a dynamic memristor array. They showed that a system with only 88 memristors is sufficient for solving a reduced 22×20 pixel MNIST handwritten digit recognition task. Hochstetter et al. [20] used nanowire networks (their junctions show memristive behavior) as reservoirs and showed that information processing in such systems is optimal when the dynamical states are a the edge of chaos. Lilak et al. [26] used atomic switch networks (which also show memristive behavior) built from silver iodide (AgI) junctions to implement reservoirs. Early work with memcapacitive devices used for RC was published by Tran et al. [48, 49]. Instead of a reservoir built from a random assembly of memristors, they used memcapacitors. Because of the lack of static power consumption, memcapacitive reservoirs are significantly more power-efficient than memristive reservoirs. Zhang et al. [53] used perovskite $NdNiO_3$ devices that can be reconfigured on demand to act as various neuromorphic building blocks. They demonstrated the capabilities of their novel devices by using a RC approach. More recently, Pei et al. [38] used oxide-based memcapacitive synapse (OMC) based on Zr-doped HfO2 (HZO) to demonstrate a power-efficient and multisensory processing reservoir computing system. The power consumption of their RC implementation outperforms most resistive reservoirs.

Biological RC. . The goal of using biological components for RC is generally to provide a computational platform that is biocompatible, allows for learning and adaptation, is ultra-low power, and that can be used on or inside a human body, and interact directly with bodily fluids, cells, and tissues. In 2020, Nguyen et al. [36] proposed a RC system that is based on random DNA strand displacement chemistry. Liu and Parhi [28] took a different approach: they used DNA memristors build from five DNA strand displacement reactions to implement a RC. More recently, Cucchi et al. [8] used organic electrochemical transistors for biosignal RC processing and Sumi et al. [44] employed culture, micropatterned biological neuronal networks as a reservoir.

Quantum and superconducting RC. The first quantum RC system was proposed by Obst et al. [37]. They used a system of Cadmium Selenide (CdSe) quantum dots for their reservoir. Fujii and Nakajima [13] exploited the natural quantum dynamics of ensemble systems for RC (in simulation). Gosh et al. [17] use numerical simulations to model a quantum reservoir with a set of fermions (e.g., quantum dots) that were arranged in a 2D lattice with random nearest-neighbor hopping. Govia et al. [19] proposed a continuous variable quantum RC based on a single nonlinear oscillator. Their results demonstrate that a simple quantum RC could be physically realized on future quantum hardware. In 2021, Rowlands et al. [42] proposed a RC based on superconducting Josephson transmission line formed by Josephson junctions. Using numerical simulations only, they showed that such circuits can do signal processing at 100 Gb/s. Suzuki et al. [45] were the first to experimentally demonstrate physical quantum RC by using IBM's superconducting quantum processors.

Photonic RC. In 2014, Vandoorne et al. [51] proposed the first integrated passive silicon photonics reservoir. Nakajima et al. [35] demonstrated a scalable on-chip photonic implementation of a RC using an integrated coherent linear photonic processor. "Photonic neuromorphic computing is of particular interest due to its significant potential for ultrahigh computing speed and energy efficiency." More recently, Liu et al. [27] demonstrated an opto-electronic synapse based on α-In2Se3 with controllable temporal dynamics that they used for multimode and multiscale reservoir computing. Their implementation is one of the few fully analog RC realizations.

3 CHALLENGES, OPEN QUESTIONS, AND NEW TRENDS

In a 2016 paper, Goudarzi and Teuscher [18] proposed a set of 11 open problems that they suggested the community should answer in order to bring RC to solve more complex real-world problems, such as real-time video processing, real-time control problems, implementing cyber-physical and embedded systems, and designing cognitive systems. Several of these questions now have answers, some have become irrelevant, and some are still open. Here, we will focus solely on (1) the hardware-related questions that are still open, (2) on new questions, (3) on new challenges that the field needs to address, and (4) on new trends.

Limited size and complexity of RC hardware. Perhaps one of the biggest issues of current material RC is the lack of large(er) scale physical implementations. To the best of our knowledge Du et al. [11] implemented the largest memristor-based RC system so far, consisting of a 32×32 crossbar array. While they were able to solve a reduced MNIST classification task, the size of the crossbar is still far too small for solving interesting real-world problems.

Optimal reservoir size and topology. It remains more of an art rather than a science to determine the necessary and sufficient size of a reservoir for a given problem. To address this challenge, some groups have proposed algorithms to incrementally grow the size and the topology of reservoirs. Qiao et al.'s [40] simulation results showed that incrementally grown reservoirs (in simulation) lead to better prediction performance and faster learning compared to fixed-size and fixed-topology reservoirs. Li and Li [25] proposed a novel approach to automatically determine the depth of a multilayer reservoirs by using a growth algorithm.

Lifelong learning. Most machine learning approaches assume a fixed, never-changing dataset that is use to train a neural network that will also be fixed. However, that is often not appropriate for real-world problems where new data is continuously added to a dataset. *Lifelong Learning* (LL) aims to develop systems that continuously learn from new data, without forgetting previously acquired knowledge. Bereska and Gavves [1] recently proposed a first method to continuously train a RC. Combined with (physically) growing reservoirs, this could be turned into a powerful platform for lifelong learning.

Alternative readout layers. The question whether more complex readout layers could benefit RC has been considered for a while, e.g., [2]. However, the approach recently emerged under a new term: *Next Generation RC* (NGRC). The basic idea of NGRC is to shift some of the non-linearity from the reservoir to the readout layers so that less data is required and fewer hyperparameters need to be optimized [16, 54]. This would make NGRC more suitable for complex tasks. NCGC would relatively straightforward to implement in hardware, however. It is an open question how much of the non-linearity should be shifted and how that will affect the learning complexity, the RC performance, and the overall hardware complexity.

Minimal architectures. It has been shown recently that randomness is not essential in reservoirs. The linear and non-linear combinations obtained from the input data can be constructed in various ways, not just by a random reservoir. The limitation, however, is that for high-dimensional and nonlinear data, the number of these combinations explodes. Ma, Prosperino, and Räth [30] showed that "[. . .] a few simple changes to the traditional reservoir computer architecture further minimizing computational resources lead to significant and robust improvements in short- and long-term predictive performances compared to similar models while requiring minimal sizes of training data sets." In their novel RC architecture, they separately fed combinations of input data separately into the reservoir, which is composed as a block-diagonal matrix of ones. The reservoir then acts as an averaging operator for the reservoir states during each update step.

While their approach simplifies the architecture and takes the randomness away, one at the same time loses a key benefit of RC for material computing: the fact that reservoirs don't need to be uniform, structured, perfect, and deterministic.

Hierarchical and modular RC. Digital systems are modular and hierarchical, forming various layers of abstraction: a set of transistors forms a logic gate, a set of logic gates forms a circuit, and circuits form the basis of architectures. Most RC systems are monolithic, which limits their computational capabilities because such networks are not easily scalable by simply increasing the number of nodes within a given reservoir. The idea of hierarchical reservoirs is not new. Triefenbach et al. [50], for example, used hierarchical reservoirs for phoneme recognition. In 2015, Bürger et al. [3] demonstrated that hierarchical reservoirs can outperform monolithic reservoir systems. More recent work also showed that hierarchical/modular reservoirs perform significantly better than monolithic reservoirs [9, 34]. As Moon et al. stated, "[w]hile software-based RC systems have broad design options such as aggressively expanding the reservoir size and inserting encoder layers between sub-reservoirs, several constraints have to be considered when designing hardware-based RC systems. For instance, physically connecting the nodes is not trivial because the complexity of routing large number of devices grows exponentially when the size of reservoir increases. Moreover, if the device is passive, active components that control the signal flows from one device to others should be also carefully designed. Due to these physical constraints, hardware-based RC systems have not shown as fast improvement in performance as software-based RC systems, even though several studies have demonstrated promising features of hardware-based RC systems such as power-efficiency and computing speed" [34].

Deep RC. The basic idea of deep RC is to use reservoirs as building blocks to create a deep, hierarchical pipeline for solving more complex tasks [14, 15]. The dynamics and training of such systems is not well understood. Actual hardware implementations remain elusive.

Multitasking RC. RC is inherent capable to solve multiple tasks concurrently. Recent work comes from Loeffler et al. [29]. They evaluated the performance of a physical neuro-memristive RC for two simultaneous tasks and showed that the structural reservoir properties play an important role.

Adaptive reservoirs. In most RC setups, the reservoir is "fixed" and left to its own, complex, nonlinear dynamics. However, it is often very beneficial to adapt the reservoir for optimal information processing. This is sometimes called "meta-learning" because the reservoir learns to learn [43]. Tuning a hardware reservoir is often rather easily possible as one has some control over the physical processes, e.g. growth parameters for nanowire networks. One could, however, also imagine that some tuning and adaption happens dynamically, as part of the training process, but perhaps on a different timescale. To the best of our knowledge, dynamic reservoir tuning has not been explored.

Materials for RC. A key question for PRC is what materials are appropriate as a reservoir. This question goes back to the beginnings of evolvable hardware [32]. A timely 2022 article by Cucci et al. [7] aims "[. . .] to give readers from fields such as material science, chemistry, or electronics an overview of implementing a reservoir computing (RC) experiment with her/his material system." Their article highlights "[. . .] the potential of RC for hardware-based neural networks, the advantages over more traditional approaches, and the obstacles to overcome for their implementation." In separate sections, and among many other relevant details, the authors provide (1) a recipe for PRC, (2) a list of properties that make up a good reservoir, and (3) a condensed summary of how to implement and train a PRC.

4 A PATH FORWARD AND A CALL FOR ACTION

Revolutionary advances in PRC require the design, synthesis, understanding, processing, and integration of advanced materials. As Cucchi et al. state: "Analog data processing using nonlinear material systems is a rapidly-growing field that is envisioned to bring about novel computational substrates and paradigms where latency and power dissipation are minimized. However, there is a considerable mismatch between the algorithmic implementation of AI on digital machines and the physical realization of physical/hardware computing networks used in material science" [7].

In the following, we identify four specific areas in which we believe progress needs to made.

Identify physical substrates. A better, more systematic approach and engineering framework to identify and evaluate complex physical materials for computation is needed. Nanotechnology bears unique opportunities to engineer, grow, and self-assemble novel physical substrates with unique properties that can potentially be harnessed for computation. We imagine creating a taxonomy of the different materials and categories that have potential for PRC.

Scaling up hardware. Composability and hierarchy are key to building large(er) PRC systems that scale up to tasks of significant complexity. All current PRC hardware is of very limited in size and complexity, which prevents solving relevant real-world tasks. How can we engineer PRC with millions or billions of parameters, perhaps of the scale of the current GPT-3 autoregressive language model, which has 175 billion parameters.

Scaling up programmability. GPT-3 was trained on hundreds of billions of words. How can PRC reach that scale (on large-scale hardware) while also providing lifelong learning capabilities?

Co-design. Computationally- and energy-efficient PRC systems are best designed by a co-design approach that involves devices, materials, interconnects, physical phenomena, algorithms, and even applications. We suggest that next generation PRC needs to rely on an integrative co-design approach that considers all levels of the computing stack.

5 CONCLUSION

For decades, the computing disciplines have relied on a a steady annual performance increase of processors. That trend has come to a rather screeching halt. Much of the focus of today's computer architecture research is geared to overcome the inefficiencies of general-purpose processors that are based on CMOS technology and the von Neumann architecture. However, the emergence of new applications and application domains has altered the requirements on computers in various ways. Neuromorphic computing and engineering is perhaps the best known example of a current application domain where we have seen great efforts over the last decade to build highly specialized and highly efficient architectures that helped to enable the recent success of artificial intelligence and machine learning applications. Specialization—with the goal to optimize performance and power/energy consumption—can happen at various levels of the entire compute stack. For the most part, such specialization involves architectures and computing paradigms beyond CMOS, beyond von Neumann, and (more) often even beyond Boolean representations. Specialization, as its name says, comes obviously at the cost of losing the ability of general-purpose computation. This is nicely illustrated in by Pieters' et al. plant RC: "[...] the results indicate that plants are not suitable for general-purpose computation but are well-suited for eco-physiological tasks such as photosynthetic rate and transpiration rate. [...] This first demonstration of physical reservoir computing with plants is key for transitioning towards a holistic view of phenotyping and early stress detection in precision agriculture applications since physical reservoir computing enables us to analyse plant responses in a general way: environmental changes are processed by plants to optimise their phenotype" [39].

Material computing has attracted increasing attention recently precisely for that reason: it allows for solving computational problems for which traditional, general-purpose architectures are not efficient. By eliminating abstraction layers and by harnessing unique physical properties of materials more directly, material computing has the potential to lead to information processing fabrics that are more computationally efficient, more power efficient, and simpler/cheaper to fabricate. In addition, PRC has great potential for physically "embedding" computing in a substrate by using the substrate itself do perform the computation. E.g., a plant-based RC could monitor an ecosystem, a DNA-based RC could monitor glucose levels in a human body, a bacterial RC could detect pathogens.

A broad success of PRC will be contingent on finding solutions to the open questions and challenges as outlined in this paper.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under award # 2318139.

REFERENCES

[1] L. Bereska and E. Gavves. 2022. Continual Learning of Dynamical Systems with Competitive Federated Reservoir Computing. *arXiv preprint arXiv:2206.13336* (2022).

[2] L. Boccato, A. Lopes, R. Attux, and F. J. Von Zuben. 2012. An extended echo state network using Volterra filtering and principal component analysis. *Neural Networks* 32 (2012), 292–302.

[3] J. Bürger, A. Goudarzi, D. Stefanovic, and C. Teuscher. 2015. Hierarchical Composition of Memristive Networks for Real-Time Computing. In *Proceedings of the 2015 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH).* IEEE Press, 33–38. https://doi.org/10.1109/NANOARCH.2015.7180583

[4] J. Bürger and C. Teuscher. 2013. Variation-tolerant Computing with Memristive Reservoirs. In *Proceedings of the 2013 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH).* 1–6. https://doi.org/10.1109/NanoArch.2013.6623028

[5] J. Cao, X. Zhang, H. Cheng, J. Qiu, X. Liu, M. Wang, and Q. Liu. 2022. Emerging dynamic memristors for neuromorphic reservoir computing. *Nanoscale* 14, 2 (2022), 289–298.

[6] J. P. Crutchfield, W. L. Ditto, and S. Sinha. 2010. Introduction to Focus Issue: Intrinsic and Designed Computation: Information Processing in Dynamical Systems—Beyond the Digital Hegemony. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 20, 3 (2010), 037101. https://doi.org/10.1063/1.3492712 arXiv:https://doi.org/10.1063/1.3492712

[7] M. Cucchi, S. Abreu, G. Ciccone, D. Brunner, and H. Kleemann. 2022. Hands-on reservoir computing: a tutorial for practical implementation. *Neuromorphic*

Computing and Engineering 2, 3 (2022), 032002. https://doi.org/10.1088/2634-4386/ac7db7

[8] M. Cucchi, C. Gruener, L. Petrauskas, P. Steiner, H. Tseng, A. Fischer, B. Penkovsky, C. Matthus, P. Birkholz, H. Kleemann, and K. Leo. 2021. Reservoir computing with biocompatible organic electrochemical networks for brain-inspired biosignal classification. *Science Advances* 7, 34 (2021), eabh0693. https://doi.org/10.1126/sciadv.abh0693 arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.abh0693

[9] M. Dale. 2018. Neuroevolution of Hierarchical Reservoir Computers. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Kyoto, Japan) *(GECCO '18)*. New York, NY, USA, 410–417. https://doi.org/10.1145/3205455.3205520

[10] M. Dale, S. Stepney, J. F. Miller, and M. Trefzer. 2017. Reservoir computing in materio: A computational framework for in materio computing. In *2017 International Joint Conference on Neural Networks (IJCNN)*. 2178–2185. https://doi.org/10.1109/IJCNN.2017.7966119

[11] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu. 2017. Reservoir computing using dynamic memristors for temporal information processing. *Nature Communications* 8, 1 (2017), 1–10.

[12] C. Fernando and S. Sojakka. 2003. Pattern Recognition in a Bucket. In *Advances in Artificial Life. Proceedings of the 7^{th} European Conference, ECAL2003 (Lecture Notes in Artificial Intelligence, Vol. 2801)*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler (Eds.). Springer-Verlag, Berlin, Heidelberg, 488–597.

[13] K. Fujii and K. Nakajima. 2017. Harnessing Disordered-Ensemble Quantum Dynamics for Machine Learning. *Phys. Rev. Applied* 8 (2017), 024030. Issue 2. https://doi.org/10.1103/PhysRevApplied.8.024030

[14] C. Gallicchio and A. Micheli. 2021. Deep Reservoir Computing. In *Reservoir Computing: Theory, Physical Implementations, and Applications*. Springer Singapore, Singapore, 77–95. https://doi.org/10.1007/978-981-13-1687-6_4

[15] C. Gallicchio, A. Micheli, and L. Pedrelli. 2017. Deep reservoir computing: A critical experimental analysis. *Neurocomputing* 268 (2017), 87–99. https://doi.org/10.1016/j.neucom.2016.12.089

[16] J. D. Gauthier, E. Bollt, A. Griffith, and W. AS. Barbosa. 2021. Next generation reservoir computing. *Nature Communications* 12, 1 (2021), 1–8.

[17] S. Ghosh, A. Opala, M. Matuszewski, T. Paterek, and T. C. H. Liew. 2019. Quantum reservoir processing. *npj Quantum Information* 5, 1 (2019), 1–6.

[18] A. Goudarzi and C. Teuscher. 2016. Reservoir Computing: Quo Vadis?. In *Proceedings of the 3^{rd} ACM International Conference on Nanoscale Computing and Communication*. 13:1–13:6. https://doi.org/10.1145/2967446.2967448

[19] L. C. G. Govia, G. J. Ribeill, G. E. Rowlands, H. K. Krovi, and T. A. Ohki. 2021. Quantum reservoir computing with a single nonlinear oscillator. *Phys. Rev. Research* 3 (Jan 2021), 013077. Issue 1. https://doi.org/10.1103/PhysRevResearch.3.013077

[20] J. Hochstetter, R. Zhu, A. Loeffler, A. Diaz-Alvarez, T. Nakayama, and Z. Kuncic. 2021. Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nature Communications* 12, 1 (2021), 1–13.

[21] H. Jaeger. 2001. *The "echo state" approach to analysing and training recurrent neural networks.* Technical Report GMD Report 148. GMD-Forschungszentrum Informationstechnik.

[22] M. Kulkarni and C. Teuscher. 2012. Memristor-based Reservoir Computing. In *Proceedings of the IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH'12)*. 226–232.

[23] R. Landauer. 1996. The physical nature of information. *Physics Letters A* 217, 4 (1996), 188–193. https://doi.org/10.1016/0375-9601(96)00453-7

[24] P. Layzell. 1998. A new research tool for intrinsic hardware evolution. In *International Conference on Evolvable Systems*. Springer, 47–56.

[25] Y. Li and F. Li. 2022. Growing deep echo state network with supervised learning for time series prediction. *Applied Soft Computing* 128 (2022), 109454. https://doi.org/10.1016/j.asoc.2022.109454

[26] S. Lilak, W. Woods, K. Scharnhorst, C. Dunham, C. Teuscher, A. Stieg, and J. Gimzewski. 2021. Spoken Digit Classification by In-Materio Reservoir Computing with Neuromorphic Atomic Switch Networks. *Frontiers in Nanotechnology* 3 (2021), 675792. https://doi.org/10.3389/fnano.2021.675792

[27] K. Liu, T. Zhang, B. Dang, L. Bao, L. Xu, C. Cheng, Z. Yang, R. Huang, and Y. Yang. 2022. An optoelectronic synapse based on α-In2Se3 with controllable temporal dynamics for multimode and multiscale reservoir computing. *Nature Electronics* (2022), 1–13.

[28] X. Liu and K. K. Parhi. 2022. DNA Memristors and Their Application to Reservoir Computing. *ACS Synthetic Biology* 11 (2022), 2202–2213. https://doi.org/doi.org/10.1021/acssynbio.2c00184

[29] A. Loeffler, R. Zhu, J. Hochstetter, A. Diaz-Alvarez, T. Nakayama, J. M Shine, and Z. Kuncic. 2021. Modularity and multitasking in neuro-memristive reservoir networks. *Neuromorphic Computing and Engineering* 1, 1 (2021), 014003. https://doi.org/10.1088/2634-4386/ac156f

[30] H. Ma, D. Prosperino, and C. Räth. 2023. A novel approach to minimal reservoir computing. *Scientific Reports* 13, 1 (2023), 12970.

[31] W. Maass, T. Natschläger, and H. Markram. 2002. Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computation* 14, 11 (11 2002), 2531–2560. https://doi.org/10.1162/089976602760407955

[32] J. F. Miller and K. Downing. 2002. Evolution in materio: Looking beyond the silicon box. In *Proceedings 2002 NASA/DoD Conference on Evolvable Hardware*. 167–176.

[33] J. F. Miller, S. L. Harding, and G. Tufte. 2014. Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence* 7, 1 (2014), 49–67.

[34] J. Moon, Y. Wu, and W. D. Lu. 2021. Hierarchical architectures in reservoir computing systems. *Neuromorphic Computing and Engineering* 1, 1 (2021), 014006. https://doi.org/10.1088/2634-4386/ac1b75

[35] M. Nakajima, K. Tanaka, and T. Hashimoto. 2021. Scalable reservoir computing on coherent linear photonic processor. *Communications Physics* 4, 1 (2021), 1–12.

[36] H. Nguyen, P. Banda, D. Stefanovic, and C. Teuscher. 2020. Reservoir Computing with Random Chemical Systems. In *Proceedings of the Artificial Life Conference*. 491–499. https://doi.org/10.1162/isal_a_00324

[37] O. Obst, A. Trinchi, S. G. Hardin, M. Chadwick, I. Cole, T. H. Muster, N. Hoschke, D. Ostry, D. Price, K. N. Pham, and T. Wark. 2013. Nano-scale reservoir computing. *Nano Communication Networks* 4, 4 (2013), 189–196. https://doi.org/10.1016/j.nancom.2013.08.005

[38] M. Pei, Y. Zhu, S. Liu, H. Cui, Y. Li, Y. Yan, Y. Li, C. Wan, and Q. Wan. 2023. Power-Efficient Multisensory Reservoir Computing Based on Zr-Doped HfO2 Memcapacitive Synapse Arrays. *Advanced Materials* (2023), 2305609. https://doi.org/10.1002/adma.202305609 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/adma.202305609

[39] O. Pieters, T. De Swaef, M. Stock, and F. Wyffels. 2022. Leveraging plant physiological dynamics using physical reservoir computing. *Scientific Reports* 12, 1 (2022), 1–14.

[40] J. Qiao, F. Li, H. Han, and W. Li. 2016. Growing echo-state network with multiple subreservoirs. *IEEE Transactions on Neural Networks and Learning Systems* 28, 2 (2016), 391–404.

[41] C. Ricciardi and G. Milano. 2022. In Materia Should Be Used Instead of In Materio. *Frontiers in Nanotechnology* 4 (2022). https://doi.org/10.3389/fnano.2022.850561

[42] G. E. Rowlands, M.-H. Nguyen, G. J. Ribeill, A. P. Wagner, L. C. G. Govia, W. A. S. Barbosa, D. J. Gauthier, and T. A. Ohki. 2021. Reservoir Computing with Superconducting Electronics. https://doi.org/10.48550/ARXIV.2103.02522

[43] A. Subramoney, F. Scherr, and W. Maass. 2021. Reservoirs Learn to Learn. In *Reservoir Computing: Theory, Physical Implementations, and Applications*, K. Nakajima and I. Fischer (Eds.). Springer Singapore, Singapore, 59–76. https://doi.org/10.1007/978-981-13-1687-6_3

[44] T. Sumi, H. Yamamoto, Y. Katori, S. Moriya, T. Konno, S. Sato, and A. Hirano-Iwata. 2022. Biological neurons act as generalization filters in reservoir computing. *arXiv preprint arXiv:2210.02913* (2022).

[45] Y. Suzuki, Q. Gao, K. C. Pradel, K. Yasuoka, and N. Yamamoto. 2022. Natural quantum reservoir computing for temporal information processing. *Scientific reports* 12, 1 (2022), 1–15.

[46] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose. 2019. Recent advances in physical reservoir computing: A review. *Neural Networks* 115 (2019), 100–123.

[47] A. Thompson. 1996. An evolved circuit, intrinsic in silicon, entwined with physics. In *International Conference on Evolvable Systems*. Springer, 390–405.

[48] D. Tran and C. Teuscher. 2019. Hierarchical Memcapacitive Reservoir Computing Architecture. In *Proceedings of the 2019 IEEE International Conference on Rebooting Computing (ICRC)*. San Mateo, CA, USA, 1–6. https://doi.org/10.1109/ICRC.2019.8914716

[49] D. J. Tran and C. Teuscher. 2021. Computational Capacity of Complex Memcapacitive Networks. *Journal on Emerging Technologies in Computing Systems* 17, 2 (2021), 1–25. https://doi.org/10.1145/3445795

[50] F. Triefenbach, A. Jalalvand, B. Schrauwen, and J.-P. Martens. 2010. Phoneme Recognition with Large Hierarchical Reservoirs. In *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta (Eds.). Curran Associates, Inc., 2307–2315.

[51] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman. 2014. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nature Communications* 5, 1 (2014), 3541. https://doi.org/doi.org/10.1038/ncomms4541

[52] A. Yoshihito. 1994. Information Processing Using Intelligent Materials — Information-Processing Architectures for Material Processors. *Journal of Intelligent Material Systems and Structures* 5, 3 (1994), 418–423. https://doi.org/10.1177/1045389X9400500315

[53] H.-T. Zhang, T. J. Park, A. N. M. N. Islam, D. S. J. Tran, S. Manna, Q. Wang, S. Mondal, H. Yu, S. Banik, S. Cheng, H. Zhou, S. Gamage, S. Mahapatra, Y. Zhu, Y. Abate, N. Jiang, S. K. R. S. Sankaranarayanan, A. Sengupta, C. Teuscher, and S. Ramanathan. 2022. Reconfigurable perovskite nickelate electronics for artificial intelligence. *Science* 375, 6580 (2022), 533–539. https://doi.org/10.1126/science.abj7943 arXiv:https://www.science.org/doi/pdf/10.1126/science.abj7943

[54] Y. Zhang and S. P. Cornelius. 2022. A Catch-22 of Reservoir Computing. *arXiv preprint arXiv:2210.10211* (2022).

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009