sensors

MDPI

Article

Colocalized Sensing and Intelligent Computing in Micro-Sensors

Mohammad H Hasan [1], Ali Al-Ramini [1], Eihab Abdel-Rahman [2], Roozbeh Jafari [3] and Fadi Alsaleem [4,*]

1 Mechanical and Materials Department, University of Nebraska–Lincoln, Lincoln, NE 68588, USA; mohammadhhasan@huskers.unl.edu (M.H.H.); aalramini@huskers.unl.edu (A.A.-R.)
2 Systems Design Engineering Department, University of Waterloo, Waterloo, ON N2L 3G1, Canada; eihab@uwaterloo.ca
3 Electrical and Computer Engineering Department, Texas A&M University, College Station, TX 77843, USA; rjafari@tamu.edu
4 Durham School of Architectural Engineering and Construction, University of Nebraska–Lincoln, Omaha, NE 68182, USA
* Correspondence: falsaleem2@unl.edu

Received: 3 September 2020; Accepted: 3 November 2020; Published: 6 November 2020

Abstract: This work presents an approach to delay-based reservoir computing (RC) at the sensor level without input modulation. It employs a time-multiplexed bias to maintain transience while utilizing either an electrical signal or an environmental signal (such as acceleration) as an unmodulated input signal. The proposed approach enables RC carried out by sufficiently nonlinear sensory elements, as we demonstrate using a single electrostatically actuated microelectromechanical system (MEMS) device. The MEMS sensor can perform colocalized sensing and computing with fewer electronics than traditional RC elements at the RC input (such as analog-to-digital and digital-to-analog converters). The performance of the MEMS RC is evaluated experimentally using a simple classification task, in which the MEMS device differentiates between the profiles of two signal waveforms. The signal waveforms are chosen to be either electrical waveforms or acceleration waveforms. The classification accuracy of the presented MEMS RC scheme is found to be over 99%. Furthermore, the scheme is found to enable flexible virtual node probing rates, allowing for up to 4× slower probing rates, which relaxes the requirements on the system for reservoir signal sampling. Finally, our experiments show a noise-resistance capability for our MEMS RC scheme.

Keywords: MEMS; reservoir computing; colocalized sensing and computing; neuromorphic computing; MEMS accelerometer

1. Introduction

1.1. Motivation

Advances in complementary metal-oxide-semiconductor (CMOS) technologies resulted in an increasing reduction in the size and cost of transistors and sensory elements. As a result, great effort is being put towards incorporating sensors and microprocessors into various devices to create 'Smart Devices'. These devices size range from large (smart cars [1,2], smart homes [3–5]) to minuscule (smart watches [6,7] and other wearable devices [8–10]). Consequently, data are being generated and collected at an ever-growing rate, with projections of continual growth. Emerging CMOS technologies have also enabled the rise of advanced robotic technologies, such as soft-robots that utilize new flexible electronic technologies, and micro-robots, which were realized by microelectromechanical systems (MEMS) sensors and fabrication techniques.

Processing data in smart devices, however, has proven to be a struggle. For example, smaller smart systems possess limited size and power capacities, which limit them to only perform simple computation locally or transmit their data to be processed in the cloud. However, improvements in computing power are plateauing as we approach the end of Moore's law due to reaching the physical limits of transistor sizes miniaturization [11] and the complicated thermal management. Therefore, there is a great need for new computing architecture suited to the challenges of smart systems. These computing architectures must:

1. Take advantage of the large amount of data generated by the sensors without overwhelming the digital processors in the systems, i.e., enabling modularity.
2. Be power- and size-efficient, cutting some sources of energy loss in this computing architecture, such as analog-to-digital converters (ADC) and memory buses.

Ideally, to avoid latencies and wireless communication costs, the proposed computing architecture should be as close to the sensory node as possible to improve performance, as shown by the potential of edge computing architectures [12]. In this paper, an analog-based colocalized sensing-and-computing architecture is presented using MEMS sensors. Computing is performed immediately at the sensor node to compress analog data into meaningful information (e.g., in an activity monitoring wearable device, acceleration data would be sent as the number of footsteps taken), which enables the production of modular, intelligent, specialized sensors that are well suited for use in intelligent systems. Here, we rely on the concept of reservoir computing (RC) as a novel computing scheme that can be incorporated using sufficiently complex nonlinear analog elements while being well suited for time-series analysis.

1.2. Literature Review

Analog computing was popularized with the introduction of neuromorphic computing by Mead [13]. Neuromorphic computing was initially defined as the use of analog circuits, namely transistors in the sub-threshold regime, rather than the typical digital operational regime, to model the dynamics of spiking neuronal networks in the human brain. The concept was later extended to include the use of digital elements and hybrid analog–digital circuits to perform the same goal [14].

Neuromorphic computing schemes have demonstrated impressive computational abilities while retaining a small footprint. A recent neuromorphic computing scheme named reservoir computing (RC), first introduced independently by Mass [15] and Jaeger [16] under the names Liquid State Machines (LSM) and Echo State Networks (ESN), respectively, stands at the forefront of non-conventional (neuromorphic) computing. In RC, large networks of nonlinear nodes with randomly chosen connection weights, named a reservoir, project input signals into a high-dimensional space to enable computation, as shown in random automata [17], quantum RCs [18,19], and arrays of spin-torque oscillators [20]. The large nonlinearity induced by the network, coupled with a forgetting echo-state property enables mapping the reservoir output into a desirable output using simple linear regression, according to the Stone–Weierstrass theorem [21]. Recently, single delay-based dynamical systems were shown to express the same dynamics as RC networks of many neuronal nodes by virtue of their 'infinite dimensionality' [22,23]. In such systems, the reservoir is composed of the states of the dynamical system at fixed time intervals. Information is retained within this system by maintaining transience in the system through input time multiplexing [22]. Delay-based RC has been demonstrated using Boolean nodes [24], photonics [25–28], spin-torque oscillators [29], coupled oscillators [30], magnet arrays [31], and recently, MEMS devices [32,33]. Among these physical devices, photonics are the most popular due to their extremely high processing speeds. However, these devices require very long fiber optic wires to incorporate delayed feedback. They also require dedicated circuits for analog-to-digital and digital-to-analog conversion, signal processing, and sampling and holding, which increases the cost of the RC system. Furthermore, the high processing speeds of photonic RC systems necessitate the use of expensive sampling elements that may be infeasible for practical implementation.

In contrast, MEMS devices can operate in the range of hundreds of Hz to MHz, allowing higher timescale and sampling flexibility in the RC scheme. Furthermore, MEMS devices are attractive for embedding a new class of delay-based RC as they can potentially perform sensing and computing co-locally, which bypasses the sensing/computing layers separation complexity inherent in traditional RC. However, performing delay-based RC using sensory elements conflicts with the requirements of input time multiplexing, as sensed input signals are typically physical quantities like acceleration which cannot be multiplexed.

1.3. Contribution and Paper Organization

In this work, we demonstrate the advantages of enabling a MEMS device to perform simultaneous sensing and computing by decoupling the reservoir inputs into physical sensory inputs and a bias time-multiplexing signal, which preserves the operation of the MEMS RC. Furthermore, we study the operation of the developed MEMS RC in a noisy environment by performing a non-trivial classification task. In particular, the contributions of this work are as follows:

1. This paper presents an RC scheme using a sensory element showing a demonstration of colocalized sensing and computing, enabling full edge computing.
2. This paper modifies the RC scheme to reduce the need for digital components. Specifically, this paper presents an approach to eliminate the need for input time multiplexing, thus reducing the need for a digital signal processor to produce the reservoir input.
3. This paper demonstrates further ability to reduce the computational costs of analog RC systems by reducing the virtual node probing rates, thus requiring less expensive sampling circuits at the reservoir output.
4. This paper experimentally tests the use of MEMS RCs in a noisy environment.

The structure of this paper is as follows: In Section 2, we describe the theory of reservoir computing, the dynamics of MEMS, and our approach of incorporating RC into the MEMS dynamics. Moreover, we describe the experimental setup used in this work in this section. In Section 3, we present our results, validating the use of the modified time-multiplexing scheme by performing a classification task using our MEMS RC. We then perform simultaneous sensing and computing using our MEMS RC scheme. Finally, in Section 4, we discuss our results and provide our conclusion.

2. Materials and Methods

2.1. Reservoir Computing

Traditional reservoir computing architectures are composed of an input signal, a reservoir of nonlinear nodes with random connection weights, and some tunable output interface, as shown in Figure 1a. When the eigenvalues of the random connection weights are below unity, the interaction between the nodes retains complexity while remaining bounded. Therefore, the input of the reservoir is projected into a higher dimensional space, enabling computation [15,16].

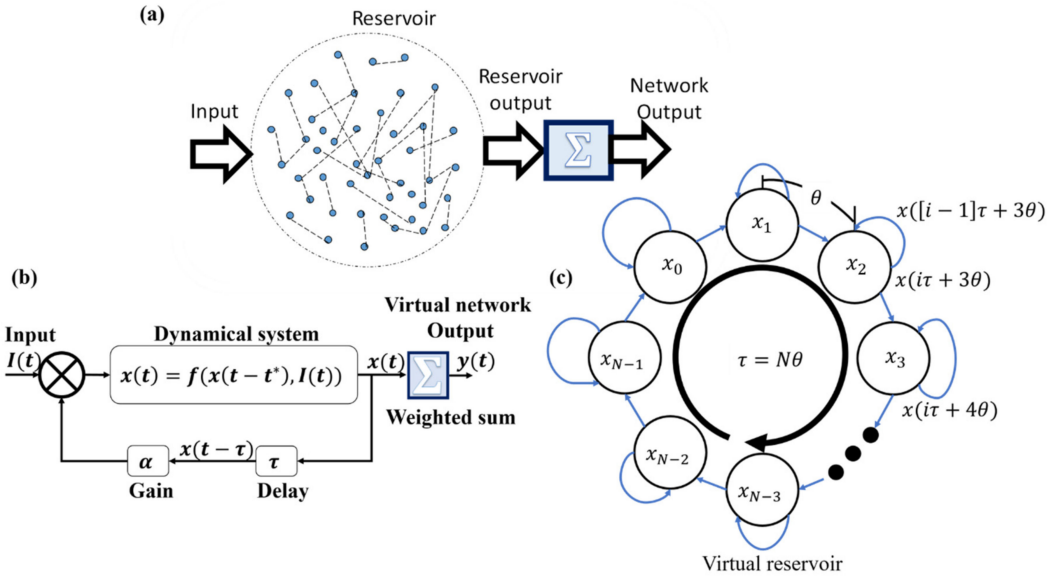

Figure 1. Reservoir computing (RC) schemes. (**a**) Classical RC utilizes a large network of randomly connected neurons. The output of the reservoir is a simple weighted sum of the neuronal response. (**b**) Delay-based RC utilizes the properties of dynamical systems and delayed feedback. (**c**) The virtual reservoir produced from delayed feedback and system dynamics.

Delay-based RC replaces the physical reservoir by a dynamical system coupled within a delayed feedback loop, as shown in Figure 1b. In this approach, instead of physical nodes, the dynamical system represents a network of N coupled virtual nonlinear nodes in time, at which the state of the dynamical system at a specific time step θ represents a virtual node. A virtual node is coupled to its preceding nodes due to the dynamical system's dependency on previous time states. Self-coupling is also enabled through delayed feedback of a gain α and a time delay $\tau = N\theta$ [22], which produces a lower triangular adjacency matrix in the system [23], as shown in Figure 1c.

The RC response of the system can be represented using a states matrix notion X, as follows:

$$X_{i,j} = x_j(i) = x((i-1)\tau + j\theta), i = 1, \ldots, T, j = 0, \ldots, N-1 \quad (1)$$

where the dynamical system state $x(t) = f(x(t-t^*), I(t))$, $f(.)$ is some update function for the dynamical system state, t is time, t^* is some past time value(s) and $I(t)$ is an external input term.

High coupling between the virtual nodes is achieved by retaining the dynamical system transience [28] through input time multiplexing [22]. Time multiplexing is usually performed using a sequence of analog-to-digital conversion, sampling-and-holding (for the duration of τ), and masking. Therefore, the reservoir input signal is given by Equation (2):

$$J^*(t) = w_j \times I((i-1)\tau), (i-1)\tau + j\theta \le t < (i-1)\tau + (j+1)\theta, i = 1, \ldots, T, j = 0, \ldots, N-1 \quad (2)$$

where $J^*(t)$ is the input time-multiplex signal for the reservoir, w_j is a periodic mask with period τ, usually chosen to be binary. The signal is a segmented piecewise constant function with a segment length of θ. $I(t)$ is the reservoir input at time t. Here, the input signal is assumed to be electrical to enable pre-processing through electrical time multiplexing. This approach, however, is ill suited for use in applications in which the input is supplied to the reservoir directly as a non-electrical signal. For instance, a MEMS accelerometer used as a reservoir cannot modulate the input acceleration waveforms it senses, as the signal measured within the MEMS device itself is non-electrical. However, the MEMS accelerometer may be used as a reservoir by using an electrically time-multiplexed

accelerometer signal from another measurement device. In this case, sensing and computing are disjointed due to the need for electrical input time multiplexing.

The output of RC is calculated by utilizing an output weight matrix W_o to generate the output Y^* that is updated at intervals of τ when all virtual nodes are processed using a weighted linear summation Equation (3):

$$Y^* = X \times W_0 \tag{3}$$

where Y^* is the RC output vector: $Y^* = \left[y_1^*,\ y_2^*,\ \ldots,\ y_T^*\ \right]^T$, $y_i^* = y^*(i\tau),\ i = 1, \ldots, T$.

The weight vector can be simply trained using linear regression, or other simple regression methods, such as Ridge regression [34].

2.2. Micro-Electro-Mechanical-System Dynamics

A doubly-cantilever, electrostatically-actuated MEMS accelerometer shown in Figure 2a,b is considered for our study. The MEMS device is considered the core of our reservoir computing scheme. The response of the MEMS accelerometer is simplified as a single-degree of freedom (SDF) spring-mass-damper system as shown in Figure 2c with dynamics governed by [35]:

$$\ddot{z}(t) + 4\pi f_n \zeta \dot{z}(t) + (2\pi f_n)^2 z(t) = \frac{\varepsilon A V_{MEMS}^2}{2m_{eff}(d-z)^2} - \ddot{y}(t) \tag{4}$$

where $z(t) = x(t) - y(t)$ is the relative displacement of the MEMS microbeam $x(t)$ with respect to the motion of the base $y(t)$ displacement at time t, f_n is the natural frequency, ζ is the damping ratio, ε is the electrical permittivity, A is the MEMS surface area, V_MEMS is the electrical voltage signal applied to the MEMS device and is usually made of a bias constant voltage V_b imposed with an AC voltage V_AC, m_eff is the effective MEMS mass, d is the nominal separation distance between the MEMS electrodes, and the dot operators represent temporal derivatives. The parameters of the used MEMS device are given in Table 1. While the dimensions are relatively large, the MEMS still exhibit the same dynamics as smaller MEMS devices due to the micro-scale d, as demonstrated in [18]. Additional information about parameter extraction and the frequency response of the MEMS accelerometer used in this paper is also available in [36]. The virtual nodes of the MEMS reservoir are represented by the state of the MEMS device $z(t)$, attained using the experimental apparatus in Figure 3.

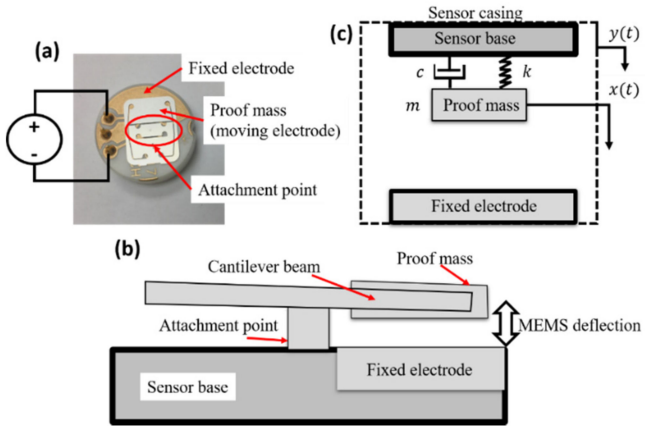

Figure 2. Microelectromechanical systems (MEMS) device schematics. (**a**) MEMS accelerometer used in this study. (**b**) Schematics of the MEMS side view showing the fixture the electrostatic gap. (**c**) Single-degree of freedom model of the MEMS accelerometer. Here, $c = 2\zeta m_{eff}(2\pi f_n\ )$ is the damping constant and $k = (2\pi f_n)^2 m_{eff}$ is the linear stiffness of the MEMS device.

Table 1. MEMS parameters.

Parameter	Physical Meaning	Value
ζ	Damping constant (in vacuum)	3×10^{-3}
f_n	MEMS natural frequency	195.3 Hz
ε	Electrical permittivity	8.85×10^{-12} F/m
A	MEMS surface area	39.6 mm^2
d	Nominal separation gap between moving and fixed electrodes	42 µm
m_{eff}	Effective MEMS mass	106 mg

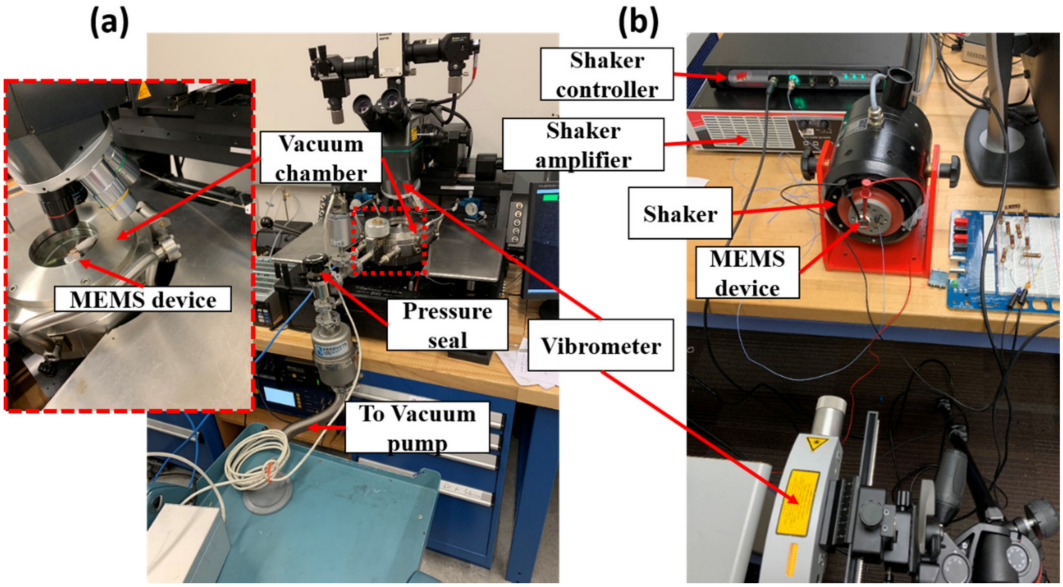

Figure 3. (**a**) Experimental setup for electrical waveform classification. The MEMS device is placed in a vacuum chamber to reduce pressure. The MEMS device serves as the reservoir in the RC system. The readout circuit is incorporated using an external data digital processor. The deflection of the MEMS device is attained by integrating the velocity signal from the laser vibrometer. (**b**) Experimental setup for acceleration waveform classification. The MEMS device is fixed on a shaker. The MEMS response is measured as the difference between the microbeam and ceramic base deflections. The shaker is controlled through a dedicated adaptive controller.

2.3. MEMS Reservoir Computing

While the traditional approach of input time multiplexing in Equation (2) can work for the conventional MEMS-based reservoir (pure computing) by modulating the input AC voltage signal [32], this approach, however, is inappropriate for our novel concept of simultaneous sensing and computing. In this new concept, the input to the device will be non-electrical in nature (acceleration, force, etc.) that cannot be modulated. To overcome this challenge, we propose to time multiplex the bias voltage as follows:

$$J(t) = w_j \times V_b, (i-1)\tau + j\theta \leq t < (i-1)\tau + (j+1)\theta, i = 1, \ldots, T, j = 0, \ldots, N-1 \quad (5)$$

Similarly to a traditional bias signal, the proposed bias time-multiplexed signal is supplied to ensure the response MEMS remains transient to facilitate node coupling, which occurs when $\theta < 1/(2\pi\zeta f_n)$. Using bias time multiplexing improves the performance of the RC by eliminating the need for sample-and-hold circuits and analog-to-digital conversion, which are otherwise necessary for input multiplexing in traditional delay-based RC. Thus, reducing the size and power consumption of the RC unit. Moreover, due to the elimination of the sample-and-hold circuit, the MEMS RC is capable of

reacting to input signals that are faster than the sampling time, τ. Traditional RCs may address this limitation by increasing the sampling frequency beyond $1/\tau$. However, increasing the sampling rate further increases the RC power requirement.

The response of the virtual nodes in the MEMS reservoir is probed by measuring the relative MEMS deflection $z(t)$ at θ intervals to construct the virtual states X matrix in Equation (1) for the RC scheme. Ideally, probing should be done slightly after the injection of the time-multiplexing signal with a small delay of δ to enable the MEMS device to respond to the new input. This is performed experimentally by measuring the MEMS response by a temporal amount δ, with $\delta < \theta$. The ideal δ chosen is determined by trial and error allowing the maximization of the RC performance.

In the next sections, we show the performance of the MEMS RC using the modified RC scheme of using for computing only. Afterward, we show the performance of the MEMS RC in a sensing-and-computing task. The computational task chosen is a binary classification signal, in which the MEMS sensor is used to distinguish between two input waveforms: rectangular waveform and triangular waveform. This task, while simple, is non-trivial [37,38]. Furthermore, it enables simple replication, especially in the sensing-and-computing task due to the ability to generate similar signals using a vibration shaker.

2.4. Experimental Procedure

The velocity response of the MEMS device is measured using laser vibrometers as shown in Figure 3a,b. The MEMS device is tested under both atmospheric pressure and reduced pressure (in a vacuum chamber) to study the effects of damping on our RC. The MEMS device is either excited using electrical signal [32] or mechanical acceleration signals $\ddot{y}(t)$. The electrical signal is fed as voltage values produced by a data acquisition device while the mechanical signal is generated using a vibration shaker as shown in Figure 3b. In both cases, the MEMS are supplied by a biasing voltage of 3 V and amplified by a 20 dB amplifier. This biasing voltage is used to produce the bias time-multiplexing signal using a binary random mask $w_j = \{0.3, 1\}$ with a 90% chance of the mask taking the value of 1. When an electrical signal classification test was performed, an additional voltage signal was also applied prior to the amplifier, representing random clumps of rectangular and triangular signals with an amplitude of 3 V. This signal is removed in the mechanical signal classification test. The mechanical acceleration inputs are applied to the MEMS device using a vibration shaker. The shaker is controlled using a dedicated adaptive controller to produce signals very close to ideal rectangular and triangular signals.

The RC reservoir parameters chosen in this work are $N = 100$, $\theta = 1$ ms and $\tau = 100$ ms. These parameters are chosen to ensure transience is maintained while decoupling the virtual nodes from the i timestep from nodes at the $i + 1$ timestep [9], while maintaining coupling between virtual nodes within the ith computing period. For tasks with low memory requirements, such as classification, recurrent connections of the virtual nodes induced by delay-feedback may be forgone with limited impact on RC performance [39]. Therefore, $\alpha = 0$ is chosen henceforth.

Acquiring the relative MEMS deflection $z(t)$ is performed by integrating $\dot{z}(t) = \dot{x}(t) - \dot{y}(t)$ after performing a high-pass filter to eliminate the low-frequency drifting often exhibited in accelerometers, where $\dot{x}(t)$ is the MEMS microbeam velocity measured at the moving electrode using a laser vibrometer and $\dot{y}(t)$ is the shaker base velocity measured by integrating an accelerometer signal. Both $\dot{x}(t)$ and $\dot{y}(t)$ are measured simultaneously and synchronized automatically using the vibration shaker controller. We note here that measuring $\dot{y}(t)$ is unnecessary when the MEMS device is actuated electrically as $\dot{z}(t) = \dot{x}(t)$ in this case.

The measured $z(t)$ is later down-sampled to a frequency of $1/\theta$ to construct the X matrix following Equation (1). For binary classification, the RC is connected to two independent classifiers (readout circuits), one for each signal type. The expected outputs of the RC scheme are generated at intervals of τ as follows:

$$Y_R = \begin{cases} 1, & \text{Rectangular input} \\ -1, & \text{Triangular input} \end{cases} \quad (6)$$

$$Y_T = \begin{cases} -1, & \text{Rectangular input} \\ 1, & \text{Triangular input} \end{cases} \quad (7)$$

where Y_R and Y_T are the expected outputs of the rectangle classifier and triangle classifier, respectively. The matrix X and the vectors $Y_R = X \times W_{oR}$ and $Y_T = X \times W_{oT}$ are split into a training set and a testing set using an 80%–20% split. The training set is used to train the actual output of the rectangle and triangle classifiers, Y_R^* and Y_T^*, by using Ridge regression [34] to optimize the output weight matrices W_{oR} and W_{oT}, respectively. Finally, the MEMS RC classification output is determined using a winner-take-all (WTA) scheme; defining success as the output of the classifier corresponding to the input waveform being higher than that of the other classifier. The success rate is calculated as:

$$\text{Success Rate} = \frac{\#\text{TestingSetCorrectClassifications}}{\#\text{SamplesInTestingSet}} \times 100\% \quad (8)$$

The pseudocode that demonstrates the process of actuating and propping the MEMS RC response is presented in Algorithm 1 (Noting that T_s is the sampling rate of velocity signals of the MEMS device and the shaker). Another pseudocode for postprocessing is presented in Algorithm 2.

Algorithm 1 Interface with MEMS device

Input N, θ
Input V_b
Input $T_s = \theta/100$
Generate $w = rand[N,1]$
Perform thresholding on w to change to binary mask
for i = 1,2,...,T **do**
for j = 1,2,...,θ **do**
Generate and Maintain $J = w[j]*V_b$ using data acquisition system
for k = 1,2,...,100 **do**
    Acquire MEMS velocity $\dot{x}$ from vibrometer
    Acquire shaker velocity $\dot{y}$ from shaker controller
  Store $\dot{x}$ into array $\dot{x}$***Array***
  Store $\dot{y}$ into array $\dot{y}$***Array***
  Wait $\boldsymbol{T_s}$
 end for
end for
end for

Algorithm 2 Postprocessing

Compute MEMS relative velocity array $\dot{z}Array = \dot{x}Array - \dot{y}Array$
Generate Expected Output array of rectangle classifier Y_R of length T
Generate Expected Output array of triangle classifier Y_T of length T
Initialize X
Input δ
Apply low-pass filter to $\dot{z}Array$
Compute $zArray$ by integrating $\dot{z}Array$
Shift $zArray$ by δ/T_s elements
Downsample with sample rate θ: $zArray \rightarrow zArrayDownSampled$
for i = 1,2,...,T **do**
    Fill X[i,ALL] = ith chunk of N elements of $zArrayDownSampled$
end for
Input $TrainSamples$
Generate $X_{Train} = X[1 : TrainSamples,ALL]$
Generate $Y_{R,Train} = Y_R[1 : TrainSamples]$
Generate $Y_{T,Train} = Y_T\ [1 : TrainSamples]$
Generate $X_{Test} = X[TrainSamples + 1 : T,ALL]$
Generate $Y_{R,Test} = Y_R[TrainSamples + 1 : T]$
Generate $Y_{T,Test} = Y_T\ [TrainSamples + 1 : T]$
Generate trained weight of rectangle classifier $W_{oR} = \left(XTrain^T\ XTrain \right)^{-1} * \left(XTrain^{-1}\ YR, Train \right)$
Generate test set results of rectangle classifier $Y_R^* = X_{Test} W_{oR}$
Generate trained weight of triangle classifier $W_{oT} = \left(XTrain^T\ XTrain \right)^{-1} * \left(XTrain^{-1}\ YT, Train \right)$
Generate test set results $Y_T^* = X_{Test} W_{oT}$
Compute classification accuracy $Success\ Rat$

3. Results

Two tests are considered: a test for MEMS as a simple computing unit and a test for MEMS colocalized sensing and computing. In both tests, we used the new concept of bias time multiplexing. Moreover, the classification tasks are performed in various pressure conditions to study the influence of damping on the performance of the RC scheme. Finally, the effect of noise is considered in these tests.

3.1. MEMS Reservoir Computing

We first test the MEMS in the worst-case scenario of operation at atmospheric pressure, resulting in high damping due to the squeeze damping effect. Operating at atmospheric pressure results in rapidly decaying transients, which may result in the development of a virtual reservoir with limited connectivity and low memory retention. The input to the MEMS device is a train of rectangle and triangle electrical signals with an amplitude equal to the bias voltage. The input voltage waveform is a random mix of periods of constant length of square and triangle signals with the same frequency and amplitude. To evaluate the response of the RC as a function of the input signal frequency, the input signal frequency is varied from 0.37%f_n to 102%f_n with a duty cycle of 50%. At each input frequency, the measured MEMS response is used to construct the virtual states matrix X by down-sampling the measured signal at θ = 1 ms.

We find that while operating the MEMS at atmospheric pressure increases squeeze film damping and may produce a shallow reservoir with sparsely connected nodes, it successfully classified input signals with relatively high frequency, as shown in Figure 4a. Low input frequencies produce a quasi-static response, which is unsuitable for the reservoir computing scheme.

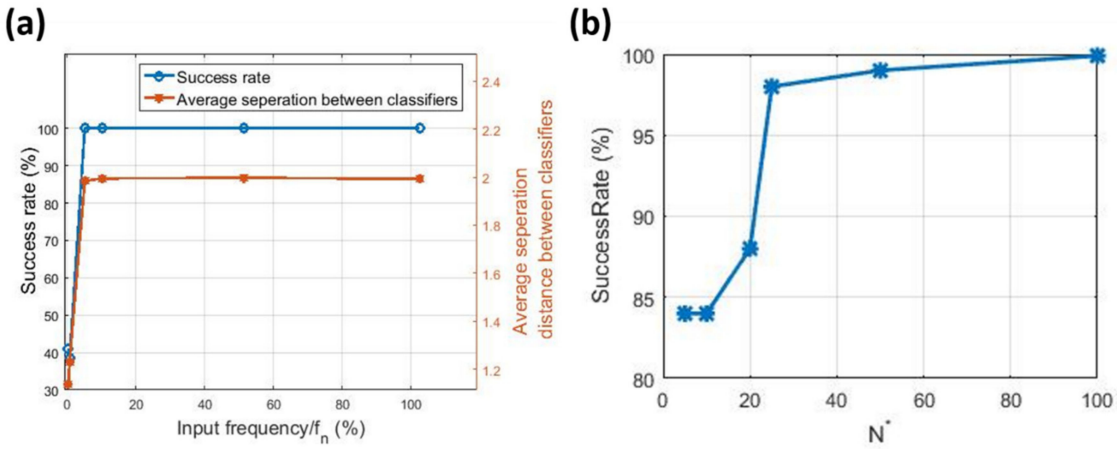

Figure 4. (**a**) Classifier accuracy as a function of the input frequency. At low input frequencies, the classifier fails to classify the input signal. However, as the frequency increases, the reservoir prediction accuracy increases to >99%. (**b**) MEMS RC classification performance at f/f_n = 20% when different N are considered.

The performance of the RC scheme can be directly inferred from the success rate. An alternative measure of performance is the average separation distance between the reservoir outputs $\overline{(X \times W_{oT} - X \times W_{oS})}$, where the bar operator represents averaging, shown as the brown line in Figure 4a. A higher average separation distance signifies a more 'confident' classification by the RC. We also test modifying the number of nodes read N^* while varying θ to maintain the same duration of computing (τ value). Figure 4b shows that $N^* = 25$ produces a 98% success rate while reducing the required sampling rate in the system by 75% ($\theta = 4$ ms). This may indicate that some of the virtual nodes obtained at a higher sampling rate ($\theta = 1$ ms) might be redundant.

The frequency-dependent success in the presented results limits the utilization of the MEMS RC to tasks with relatively high-frequency input. To extend the operating frequency to quasi-static

inputs, the memory of the system needs to be improved by reducing the squeeze film damping. This is achieved by reducing the operating pressure. Towards this end, the MEMS RC is placed in a vacuum chamber as shown in Figure 3a and is driven by an electrical signal at $0.37\% f_n$. Two types of variability are introduced separately to investigate the RC performance in classifying low-frequency signals in the presence of noise: (1) colored noise due to a combination of low-frequency ambient noise and ground vibrations resulting from running the vacuum pump, and (2) parameter drift as pressure built up in the vacuum chamber after shutting off the vacuum pump ahead of the experiment. Despite quasi-static input and the introduction of noise, the MEMS RC is capable of performing successful classification (pressure variation noise (99.8%, Figure 5a) and colored noise (99.66%, Figure 5b)). Importantly, compared to Figure 4, Figure 5 shows that the classification success rate of the MEMS RC when operated under reduced pressure improves its operation at the low input frequency.

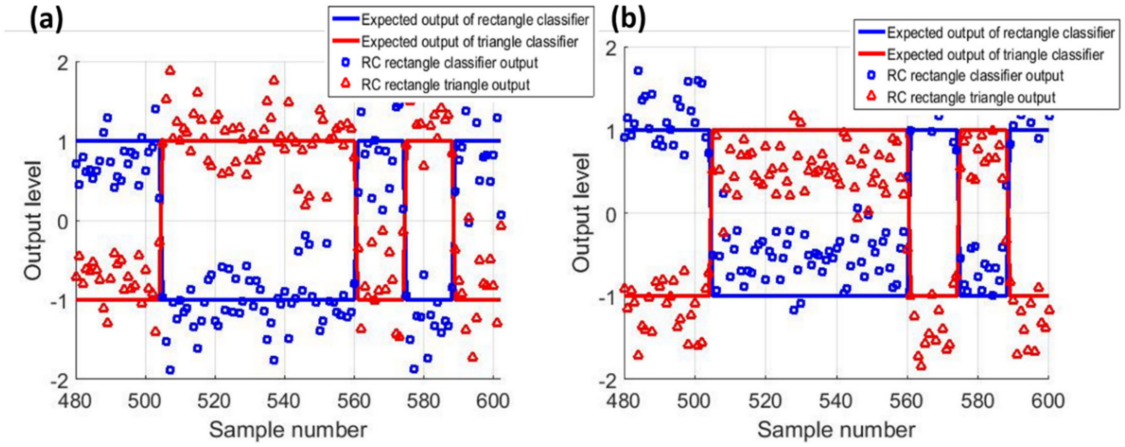

Figure 5. Experimental classification of low-frequency signal using the MEMS RC (**a**) under parameter (pressure) drift (**b**) and colored noise. The real-time results of the RC show success rates of 99.8% and 99.66%, respectively.

3.2. MEMS Reservoir Sensing and Computing Unit

The main advantage of using MEMS as an RC is the ability to perform sensing and computing simultaneously. In this case, MEMS can directly extract complex information from its environment. As an example, we present the first demonstration of a MEMS accelerometer that can directly classify a train of acceleration signals. The acceleration train is generated using a closed-loop controlled shaker, Figure 3b. The sensing and computing task is performed experimentally using an acceleration pulse train with a frequency of $1.1367 f_n$ and an amplitude of 5 g, Figure 6a. The frequency is the smallest frequency capable of producing sufficiently high acceleration to enable MEMS motion under the influence of squeeze-film damping at atmospheric pressure. The MEMS device is forced into transience using a time-multiplexed signal, which modulates the biasing voltage rather than the input signal. The relative displacement of the MEMS device is shown in Figure 6b. Figure 6c shows the MEMS RC response sampled and held each 1 ms, 80% of the data points are used to create the training set, and 20% of the data points are used for testing. After training, the MEMS network successfully classifies 99.6% of the testing data, which is on par with similar RC schemes for this test [31,37,38] and on par with the results observed from Figure 4. The success of this scheme shows that the sensing and computing using RC is possible using a slight modification to the input time-multiplexing scheme. We note here that the similarity between the results from the MEMS computing and the MEMS sense-and-compute tasks suggests that operation in a vacuum may still be required in colocalized sensing and computing tasks for low-frequency acceleration signals.

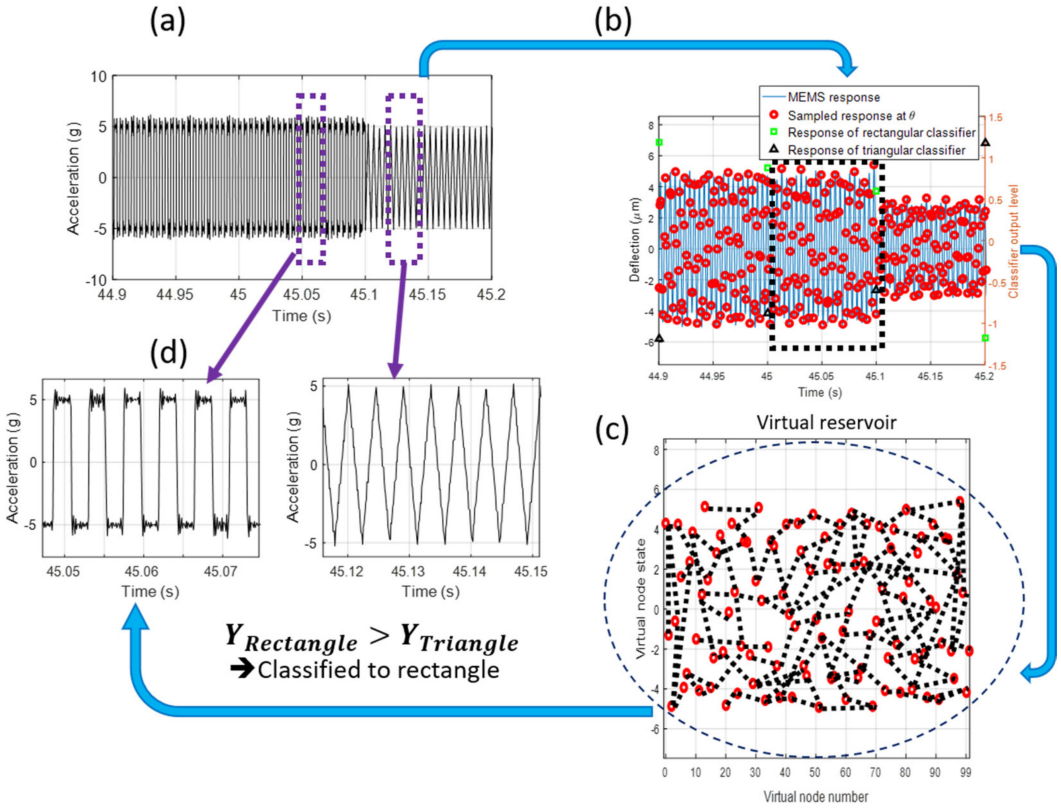

Figure 6. MEMS sense-and-compute scheme: (**a**) a sample of the acceleration signal generated by the shaker. (**b**) The response of the MEMS device and virtual node extraction. (**c**) Visualization of the virtual reservoir within the network. The response of the virtual nodes shown in this figure is used to compute the reservoir output after all the virtual nodes are updated. (**d**) Classification process based on the RC output.

4. Discussion and Conclusions

The use of input-based time multiplexing was previously shown to be beneficial for implementing RCs using a single physical dynamical node. However, this scheme requires injecting data into the reservoir by using a combination of external sensory elements and computationally inefficient analog-to-digital converters (ADCs) to enable input time multiplexing and digital-to-analog converters (DACs) to supply the input signal into the analog reservoir. These external components can be eliminated by using sensory elements, such as colocalized sensing-and-computing nodes, where information is simultaneously gathered and processed, as shown in Figure 7. This process can be achieved by utilizing a bias voltage signal for time multiplexing rather than the input signal, thus enabling the RC to immediately perform computing on environmental signals (acceleration in this work) without compromising computational performance. Because the bias signal is time-multiplexed, rather than the input signal, the analog-to-digital conversion and digital modulation may be eliminated and replaced by an analog input time-multiplexing circuit with a fixed output. Delay-feedback may also be performed using analog electronics using capacitive circuits. Furthermore, amplification can be performed passively by relying on the mechanical and electrical resonances in MEMS devices, as was demonstrated in previous works [40]. Thus, the pre-processing and processing steps of the reservoir computer can be performed entirely in an analog fashion, which will be the subject of our future work.

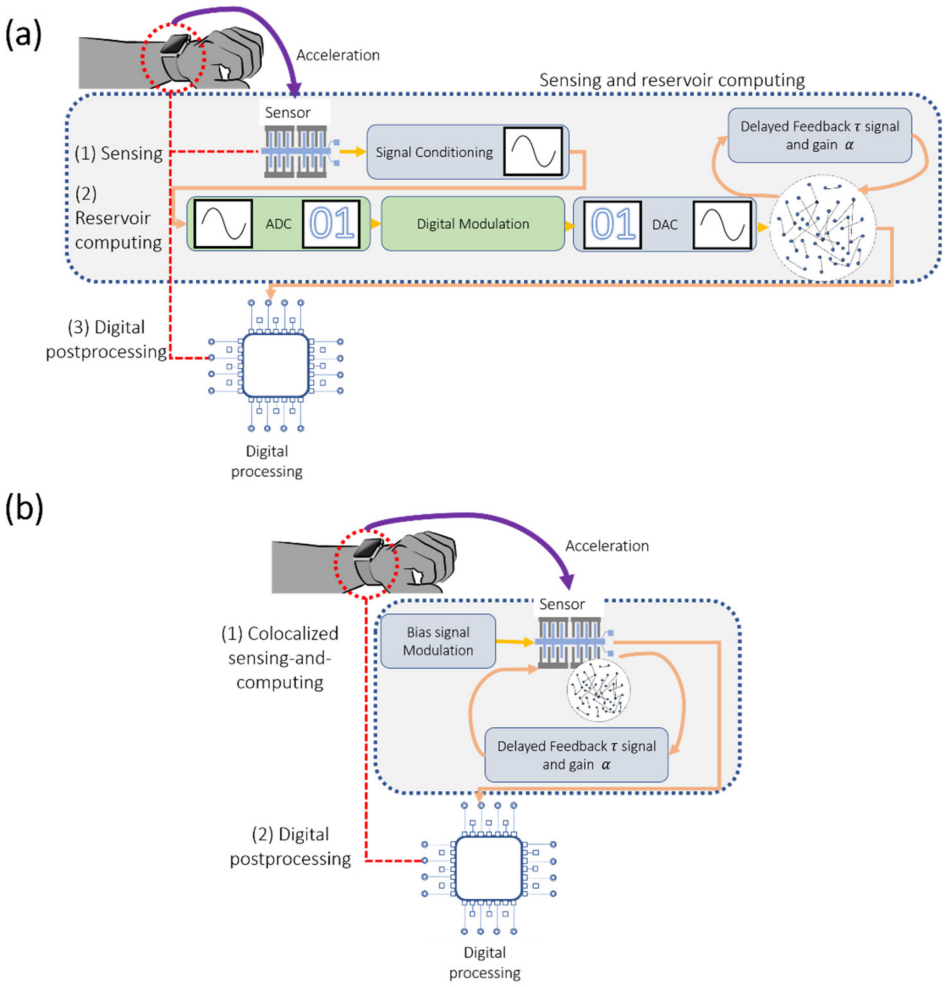

Figure 7. Illustration demonstrating the size reduction of the RC scheme with the use of sensors for colocalized sensing-and-computing. In this figure, digital components are represented by green blocks. Analog components are represented by grey components. (**a**) Delay RC scheme with separated sensing and computing. (**b**) Delay RC scheme with colocalized sensing and computing.

This approach is also shown to be resilient to different types of noise and successful even under high damping conditions. Moreover, it enables reducing the virtual node-probing period from θ to as high as 4θ, which can also improve the efficiency of the scheme by reducing the reservoir sampling rate.

The utilization of MEMS devices in the modified RC scheme enables sensing and computing. Additional benefits are expected when simultaneously targeting and exciting multiple mode shapes of a MEMS microstructure. In this case, the MEMS device is modeled as a contentious beam rather than a simple SDF model, which may enable the production of multiple independent virtual reservoirs with different timescales within the same MEMS device. Such a concept of using multiple reservoirs to boost the RC performance [41], especially asynchronous reservoirs [42], is recently shown to circumvent the performance plateau as the number of virtual nodes increases.

While previous works have shown the potential of performing simple edge computing using MEMS sensors [43], this approach shows a new generation of intelligent sensors can be produced without using large networks of sensory elements [44]. Such sensors will entirely utilize transience, increasing their speeds. Furthermore, these sensors may be capable of performing edge computation by performing classification and/or prediction. For example, RC sensors may be capable of compensating for interference due to measurement in nonlinear systems, such as compensation for flow changes due to the insertion of a flow rate sensor into a water pipe.

In conclusion, in this work we demonstrated a reservoir computing scheme using a single MEMS sensor to perform colocalized sensing and computing to reduce the cost of RC implementation. The developed scheme modifies the approach used in traditional RC systems by eliminating the need for input time multiplexing, which reduces the need for digital electronics at the reservoir inputs to perform sample-and-hold, analog-to-digital conversion, digital processing and digital-to-analog conversion. Instead, the MEMS RC scheme utilizes a time-multiplexed bias signal (modulated bias) that may be generated using simpler components. The MEMS RC scheme was tested through solving a waveform classification problem, in which this waveform is provided either electrically (computing task) or as an acceleration signal (sensing and computing). In both cases, the MEMS RC yielded a classification accuracy > 99%, even in the presence of noise. Finally, this paper further reduces the cost of using RC by showing an acceptable (~98%) classification accuracy at reduced virtual node probing rates (up to 4x slower virtual node probing), reducing the required MEMS signal sampling rate.

Author Contributions: M.H.H. performed the experimental work involving MEMS reservoir computing, post-processed the data for reservoir computing and wrote the manuscript draft, A.A-R. performed the experimental work for MEMS colocalized sensing and computing, E.A.-R. supervised the experimental work, R.J. and F.A. supervised this work and provided edits to the manuscript. Conceptualization, F.A. and M.H.H.; methodology, M.H.H.; software, M.H.H.; validation, M.H.H. and F.A.; formal analysis, M.H.H.; investigation, M.H.H.; resources, F.A. and E.A.-R.; data curation, M.H.H. and A.A.-R.; writing—original draft preparation, M.H.H.; writing—review and editing, F.A., E.A.-R., R.J.; visualization, M.H.H.; supervision, F.A. and R.J.; project administration, F.A.; funding acquisition, F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by NSF grants (#1935598, #1935641). Experimental work on the classification of electrical input waveforms was supported by the University of Waterloo, Waterloo, Canada, under IRPG grant "Micro- Neuro Computing".

Acknowledgments: We thank Mohamed Arabi from the University of Waterloo for helping with the experimental setup in this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cao, J.; Song, C.; Song, S.; Peng, S.; Wang, D.; Shao, Y.; Xiao, F. Front vehicle detection algorithm for smart car based on improved SSD model. *Sensors* **2020**, *20*, 4646. [CrossRef] [PubMed]
2. Wang, X.; Wu, C.; Xue, J.; Chen, Z. A Method of Personalized Driving Decision for Smart Car Based on Deep Reinforcement Learning. *Information* **2020**, *11*, 295. [CrossRef]
3. Ding, J.; Wang, Y. A WiFi-based Smart Home Fall Detection System using Recurrent Neural Network. *IEEE Trans. Consum. Electron.* **2020**. [CrossRef]
4. Lin, Y.H. A Parallel Evolutionary Computing-Embodied Artificial Neural Network Applied to Non-Intrusive Load Monitoring for Demand-Side Management in a Smart Home: Towards Deep Learning. *Sensors* **2020**, *20*, 1649. [CrossRef] [PubMed]
5. Ng, W.W.; Xu, S.; Wang, T.; Zhang, S.; Nugent, C. Radial Basis Function Neural Network with Localized Stochastic-Sensitive Autoencoder for Home-Based Activity Recognition. *Sensors* **2020**, *20*, 1479. [CrossRef] [PubMed]
6. Ashry, S.; Ogawa, T.; Gomaa, W. CHARM-Deep: Continuous Human Activity Recognition Model Based on Deep Neural Network using IMU Sensors of Smartwatch. *IEEE Sens. J.* **2020**, *20*, 8757–8770. [CrossRef]
7. Sultana, M.; Al-Jefri, M.; Lee, J. Using Machine Learning and Smartphone and Smartwatch Data to Detect Emotional States and Transitions: Exploratory Study. *JMIR mHealth uHealth* **2020**, *8*, e17818. [CrossRef]
8. Shen, S.; Gu, K.; Chen, X.R.; Lv, C.X.; Wang, R.C. Gesture Recognition Through sEMG with Wearable Device Based on Deep Learning. *Mob. Netw. Appl.* **2020**. [CrossRef]
9. Bartlett, M.D.; Markvicka, E.J.; Majidi, C. Rapid fabrication of soft, multilayered electronics for wearable biomonitoring. *Adv. Funct. Mater.* **2016**, *26*, 8496–8504. [CrossRef]
10. Markvicka, E.; Wang, G.; Lee, Y.C.; Laput, G.; Majidi, C.; Yao, L. ElectroDermis: Fully Untethered, Stretchable, and Highly-Customizable Electronic Bandages. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, Glasgow, UK, 4–9 May 2019. [CrossRef]
11. Waldrop, M.M. The chips are down for Moore's law. *Nat. News* **2016**, *530*, 144. [CrossRef]

12. Shi, W.; Dustdar, S. The promise of edge computing. *Computer* **2016**, *49*, 78–81. [CrossRef]
13. Mead, C. Neuromorphic electronic systems. *Proc. IEEE* **1990**, *78*, 1629–1636. [CrossRef]
14. Schuman, C.D.; Potok, T.E.; Patton, R.M.; Birdwell, J.D.; Dean, M.E.; Rose, G.S.; Plank, J.S. A survey of neuromorphic computing and neural networks in hardware. *arXiv* **2017**, arXiv:1705.06963.
15. Maass, W.; Natschläger, T.; Markram, A.H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Comput.* **2002**, *14*, 2531–2560. [CrossRef]
16. Jaeger, H. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Ger. Natl. Res. Cent. Inf. Technol. GMD Tech. Rep.* **2001**, *148*, 13.
17. Snyder, D.; Goudarzi, A.; Teuscher, C. Computational capabilities of random automata networks for reservoir computing. *Phys. Rev. E* **2013**, *87*, 042808. [CrossRef]
18. Fujii, K.; Nakajima, K. Harnessing disordered-ensemble quantum dynamics for machine learning. *Phys. Rev. Appl.* **2017**, *8*, 024030. [CrossRef]
19. Nakajima, K.; Fujii, K.; Negoro, M.; Mitarai, K.; Kitagawa, M. Boosting computational power through spatial multiplexing in quantum reservoir computing. *Phys. Rev. Appl.* **2019**, *11*, 034021. [CrossRef]
20. Kanao, T.; Suto, H.; Mizushima, K.; Goto, H.; Tanamoto, T.; Nagasawa, T. Reservoir Computing on Spin-Torque Oscillator Array. *Phys. Rev. Appl.* **2019**, *12*, 024052. [CrossRef]
21. Konkoli, Z. *Advances in Unconventional Computing*; Springer: Cham, Switzerland, 2017. [CrossRef]
22. Appeltant, L.; Soriano, M.C.; Sande, G.V.D.; Danckaert, J.; Massar, S.; Dambre, J.; Schrauwen, B.; Mirasso, C.R.; Fischer, A.I. Information processing using a single dynamical node as complex system. *Nat. Commun.* **2011**, *2*, 468. [CrossRef]
23. Hart, J.D.; Larger, L.; Murphy, T.E.; Roy, A.R. Delayed dynamical systems: Networks, chimeras and reservoir computing. *Philos. Trans. R. Soc. A* **2019**, *377*, 20180123. [CrossRef]
24. Haynes, N.D.; Soriano, M.C.; Rosin, D.P.; Fischer, I.; Gauthier, A.D.J. Reservoir computing with a single time-delay autonomous Boolean node. *Phys. Rev. E* **2015**, *91*, 020801. [CrossRef] [PubMed]
25. Antonik, P.; Haelterman, M.; Massar, S. Brain-inspired photonic signal processor for generating periodic patterns and emulating chaotic systems. *Phys. Rev. Appl.* **2017**, *7*, 054014. [CrossRef]
26. Brunner, D.; Penkovsky, B.; Marquez, A.; Jacquot, M.; Fischer, I.; Larger, A.L. Tutorial: Photonic neural networks in delay systems. *J. Appl. Phys.* **2018**, *124*, 152004. [CrossRef]
27. Kitayama, K.I.; Notomi, M.; Naruse, M.; Inoue, K.; Kawakami, S.; Uchida, A.A. Novel frontier of photonics for data processing—Photonic accelerator. *APL Photonics* **2019**, *4*, 09090. [CrossRef]
28. Hermans, M.; Antonik, P.; Haelterman, M.; Massar, S. Embodiment of learning in electro-optical signal processors. *Phys. Rev. Lett.* **2016**, *117*, 128301. [CrossRef] [PubMed]
29. Riou, M.; Torrejon, J.; Garitaine, B.; Araujo, F.A.; Bortolotti, P.; Cros, V.; Tsunegi, S.; Yakushiji, K.; Fukushima, A.; Kubota, H.; et al. Temporal pattern recognition with delayed-feedback spin-torque nano-oscillators. *Phys. Rev. Appl.* **2019**, *12*, 024049. [CrossRef]
30. Velichko, A.A.; Ryabokon, D.V.; Khanin, S.D.; Sidorenko, A.V.; Rikkiev, A.G. Reservoir computing using high order synchronization of coupled oscillators. *arXiv* **2020**, arXiv:2004.04114.
31. Zhou, P.; McDonald, N.R.; Edwards, A.J.; Loomis, L.; Thiem, C.D.; Friedman, J.S. Reservoir Computing with Planar Nanomagnet Arrays. *arXiv* **2020**, arXiv:2003.10948.
32. Dion, G.; Mejaouri, S.; Sylvestre, A.J. Reservoir computing with a single delay-coupled non-linear mechanical oscillator. *J. Appl. Phys.* **2018**, *124*, 152132. [CrossRef]
33. Barazani, B.; Dion, G.; Morissette, J.F.; Beaudoin, L.; Sylvestre, J. Microfabricated Neuroaccelerometer: Integrating Sensing and Reservoir Computing in MEMS. *J. Microelectromechan. Syst.* **2020**, *29*, 338–347. [CrossRef]
34. Hoerl, A.E.; Kennard, A.R.W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* **1970**, *12*, 55–67. [CrossRef]
35. Younis, M. *MEMS linear and Nonlinear Statics and Dynamics*; Springer Science & Business Media: New York, NY, USA, 2011. [CrossRef]
36. Alsaleem, F.M.; Younis, M.I.; Ouakad, H.M. On the nonlinear resonances and dynamic pull-in of electrostatically actuated resonators. *J. Micromech. Microeng.* **2009**, *19*, 045013. [CrossRef]
37. Paquot, Y.; Duport, F.; Smerieri, A.; Schrauwen, B.; Haelterman, M.; Massar, S. Optoelectronic reservoir computing. *Sci. Rep.* **2012**, *2*, 287. [CrossRef]

38. Vandoorne, K.; Dierckx, W.; Schrauwen, B.; Verstraeten, D.; Baets, R.; Bienstman, P.; Campenhout, A.V.J. Toward optical signal processing using photonic reservoir computing. *Opt. Express* **2008**, *16*, 11182–11192. [CrossRef]
39. Hicke, K.; Escalona-Morán, M.A.; Brunner, D.; Soriano, M.C.; Fischer, I.; Mirasso, C.R. Information processing using transient dynamics of semiconductor lasers subject to delayed feedback. *IEEE J. Sel. Top. Quantum Electron.* **2013**, *19*, 1501610. [CrossRef]
40. Jaber, N.; Hafiz, M.A.A.; Kazmi, S.; Hasan, M.; Alsaleem, F.; Ilyas, S.; Younis, M. Efficient excitation of micro/nano resonators and their higher order modes. *Sci. Rep.* **2019**, *9*, 319. [CrossRef]
41. Pathak, J.; Hunt, B.; Girvan, M.; Lu, Z.; Ott, E. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.* **2018**, *120*, 024102. [CrossRef] [PubMed]
42. Penkovsky, B.; Porte, X.; Jacquot, M.; Larger, L.; Brunner, D. Coupled nonlinear delay systems as deep convolutional neural networks. *Phys. Rev. Lett.* **2019**, *123*, 054101. [CrossRef] [PubMed]
43. Abbasalipour, A.; Nikfarjam, H.; Pourkamali, S. An 8-Bit Digitally Operated Micromachined Accelerometer. *J. Microelectromechanical Syst.* **2020**, *29*, 1132–1136. [CrossRef]
44. Rafaie, M.; Hasan, M.H.; Alsaleem, F.M. Neuromorphic MEMS sensor network. *Appl. Phys. Lett.* **2019**, *114*, 163501. [CrossRef]