



Article

Spiking Neural Networks for Structural Health Monitoring

George Vathakkattil Joseph ^{1,†}  and Vikram Pakrashi ^{2,*,†} 

UCD Centre for Mechanics, Dynamical Systems and Risk Laboratory, School of Mechanical and Materials Engineering, University College Dublin, 4 Dublin, Ireland

* Correspondence: vikram.pakrashi@ucd.ie; Tel.: +353-1-716-1833

† These authors contributed equally to this work.

Abstract: This paper presents the first implementation of a spiking neural network (SNN) for the extraction of cepstral coefficients in structural health monitoring (SHM) applications and demonstrates the possibilities of neuromorphic computing in this field. In this regard, we show that spiking neural networks can be effectively used to extract cepstral coefficients as features of vibration signals of structures in their operational conditions. We demonstrate that the neural cepstral coefficients extracted by the network can be successfully used for anomaly detection. To address the power efficiency of sensor nodes, related to both processing and transmission, affecting the applicability of the proposed approach, we implement the algorithm on specialised neuromorphic hardware (Intel® Loihi architecture) and benchmark the results using numerical and experimental data of degradation in the form of stiffness change of a single degree of freedom system excited by Gaussian white noise. The work is expected to open a new direction of SHM applications towards non-Von Neumann computing through a neuromorphic approach.

Keywords: spiking neural network; neuromorphic; low-power; Loihi; cepstrum; Mahalanobis distance



Citation: Joseph, G.V.; Pakrashi, V. Spiking Neural Networks for Structural Health Monitoring. *Sensors* **2022**, *22*, 9245. <https://doi.org/10.3390/s22239245>

Academic Editor: Mohammad Noori

Received: 1 October 2022

Accepted: 23 November 2022

Published: 28 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Effective structural health monitoring (SHM) requires acquisition, transmission, and processing of multidimensional data over long periods of time. These typically comprise of acceleration, displacement, velocity, and strain data, along with environmental factors such as temperature [1,2]. To address the needs of acquiring high-resolution, high-volume data from a structure, distributed sensor networks are becoming popular [3]. Wireless sensor networks are favoured since they allow significant simplification of instrumentation in practice, and other benefits such as minimal invasion of host structure and remote reconfigurability [4]. Powering the sensor nodes, however, poses a challenge. Wiring for power from a grid partially offsets the advantages of wireless operation. Moreover, structures are often situated in remote or harsh areas with inadequate access to sources of electricity. Reliably powering such remote sensor nodes with appropriate processing capacity, along with information transmission, converts the problem to an energy management issue as well. This aspect consequently impacts all system design choices from sensor network topology, transmission rate, and on-node/off-node processing, amongst others [5].

Energy harvesters based on solar, thermal, wind, piezoelectric, or electromagnetic effects are being extensively studied to establish them as power source candidates for such scenarios [6]. The limited power available, however, from such devices is generally rationed for the acquisition and transmission phases, with the majority of the processing relegated to a central hub or off-site laboratory where access to sufficient processing power is available [5]. An optimisation problem of transmission is encountered here because Internet of Things (IoT)-focused networks such as LoRaWAN and Narrowband-IoT suffer from low bandwidth, whereas Wi-Fi suffers from short range [7–9]. Ideally, if damage-sensitive features (DSFs) [10] could be extracted from the data at the nodes themselves, the transmission required would be reduced to the feature vectors or their changes [11].

Methods and algorithms oriented towards this approach, as well as using the harvester itself as a sensor, have been reported [12–16]. However, currently, only basic processing can be achieved at the node with the energy generated by harvesters. Nevertheless, pattern recognition algorithms for SHM that adopt advances in machine learning have been proposed, with promising initial results [17–21]. The ability to implement such algorithms with limited power could prove to be a breakthrough in SHM technology.

A typical example of basic processing is transformation of a sensor input to the frequency domain. Even this transformation can be energy intensive, even with a fast Fourier transform (FFT) algorithm. However, the architecture used by the human ear for the qualitatively similar operation of spectral decomposition is extremely efficient. A linear scaling (N) in time and power is achieved [22] compared to an $N \log N$ scaling for the FFT, where N is the number of output frequency bins. With this context, we consider neuromorphic computing, which is a relatively novel paradigm of hardware and software design implementing functional characteristics of such biological architectures on silicon [23,24]. The approach is especially suited to processes where a loss in precision is tolerable in exchange for greater energy efficiency and/or speed. SHM using distributed sensor nodes is a perfect candidate for a neuromorphic approach due to the demand for low-power computing of DSFs at the node level.

This paper shows the extraction of DSFs using a novel neuromorphic hardware platform and proposes its use in practice. The DSF vector is composed of coefficients generated using an approximate version of the power-cepstrum [25,26]. The cepstrum transformation has been long established as a method in condition monitoring of rotary machines [27,28]. The conceptual similarity of the transform to the processing in the human ear has led to its early adoption in audio-processing applications as well [29,30]. Application of cepstral analysis to SHM of civil infrastructure was proposed by a few authors recently [20,31–35].

It is observed that a neuromorphic approach can have a particular advantage in terms of computational and energy parsimony, while offering high pattern recognition capabilities [36–38]. A neuromorphic implementation of cepstral analysis for speech recognition and synthesis was shown recently [39]. As a first example of its kind, this paper demonstrates a neuromorphic implementation of cepstral coefficient extraction for SHM and implements it on the Intel[®] Loihi neuromorphic platform [40]. The efficacy and power consumption is reported, showing promise for further investigation. While Loihi is still far from being a low-cost edge processor, SNN implementations in SHM, demonstration of solutions, and development of benchmarks of results create a pathway towards reaching this goal.

Recent proposals and simulations of utilising spiking neural networks for damage detection [41,42] showcase the potential of learning damage-sensitive features through backpropagation. However, a recurring challenge in the deployment of neural-network-based solutions in safety-critical applications is the explainability of the neural network model applied. In this work, an alternative approach is adopted where a minimal, yet well understood, feature extraction and anomaly detection method is converted to a spiking neural network. An experimental evaluation is performed on the neuromorphic platform Loihi to profile and validate the functionality. This demonstration could serve to lower the barrier to adoption of such technology for applications in SHM in different sectors [43,44].

2. Theory

Neurobiological systems are complex, and the field of neuroscience has made strides in understanding the functional characteristics of the core components that allow sensing and computation in the brain. This understanding has led to the advent of neuromimetic and neuromorphic devices where electronic components that capture these functional characteristics reproduce behaviour similar to the brain. To simplify the process of neuromorphic system design, multiple mathematical formalisms have been developed, allowing high levels of abstraction while preserving core aspects of behaviour. This work utilises

the Neural Engineering Framework (NEF) formalism since it allows the construction of a spiking neural network of a **known transform** through an optimisation procedure as opposed to frameworks more suitable for learning a transformation based on input–output data. A brief summary of the formalism is outlined here.

2.1. Neural Engineering Framework

Encoding/decoding, transmission, and processing of information in the human brain is predominantly carried out through spiking activity of neurons. The neural engineering framework (NEF) provides a set of principles intended as a practical method for non-neuroscientists to transform traditional algorithms to a neuromorphic architecture using spiking (or nonspiking) neural networks [45]. The NEF allows the mapping of both linear and nonlinear dynamical systems to SNNs [46]. We use the NEF to build our network to extract the cepstrum. Spiking neural networks (SNNs) described here are distinct from the field of artificial neural networks, which has recently gained mainstream adoption [47]. There is no fitting or learning of weights from data involved in this application. The SNN in our case is fully defined based on the transformations to be performed (SNNs can be considered as a generalisation of conventional ANN and learning can be performed as well).

The three guiding principles of the NEF, pertaining to representation, transformation, and dynamics, are summarised below. A formal description of the principles is provided in Appendix A.

2.1.1. Representation

A continuous time-varying signal of arbitrary dimensionality is to be represented by a population of spiking neurons. To achieve this, a nonlinear encoding and linear decoding scheme is used. Each neuron model is characterised by a tuning curve which defines its spiking activity as a function of its input current. Consider a signal, as shown in Figure 1a. The signal is to be encoded, for example, using eight neurons (representation error reduces with larger populations of neurons). The eight neurons are divided equally into those that spike due to positive signals and those that spike due to negative signals. The tuning curves of the eight neurons are shown in Figure 1b. The tuning curves have equally spaced intercepts in the range $(-0.9, 0.9)$. Based on the intercepts, the spiking activity starts and increases with signal amplitude in both cases. The spiking activity $a(J)$ in each of the eight neurons representing the input signal is shown in Figure 1c. Note how the neurons with higher intercepts, such as neuron 7, start firing only at a high signal level. Thus, to represent a signal vector \mathbf{x} , the corresponding current levels \mathbf{J} driving the neurons can be determined as:

$$\mathbf{J}(\mathbf{x}) = \alpha \mathbf{e} \cdot \mathbf{x} + \mathbf{J}_{\text{bias}} \quad (1)$$

where \mathbf{e} are unit vectors (called encoders) that specify the direction of spiking (positive or negative) for each neuron, α is a scaling factor, and \mathbf{J}_{bias} determines the intercept or the level at which spiking starts/stops (in Figure 1b, the lines are not strictly straight since the neuron model is “leaky”. See Appendix A for more details).

Decoding of the spiking activity is performed using a weighted sum of the spiking activity of all neurons:

$$\hat{\mathbf{x}} = \sum_{i=0}^{n-1} \mathbf{d}_i a_i \quad (2)$$

where $\hat{\mathbf{x}}$ is the decoded estimate of \mathbf{x} , \mathbf{d}_i is the set of decoding weight for each neuron, and a_i is the spiking activity of the neuron. \mathbf{d} is, in general, calculated by solving the least squares problem:

$$\mathbf{A} \mathbf{d} = \mathbf{X} \quad (3)$$

where \mathbf{A} is the activity of all neurons for all inputs and \mathbf{X} is a sample input. With this representation scheme, any smooth signal can be encoded and decoded.

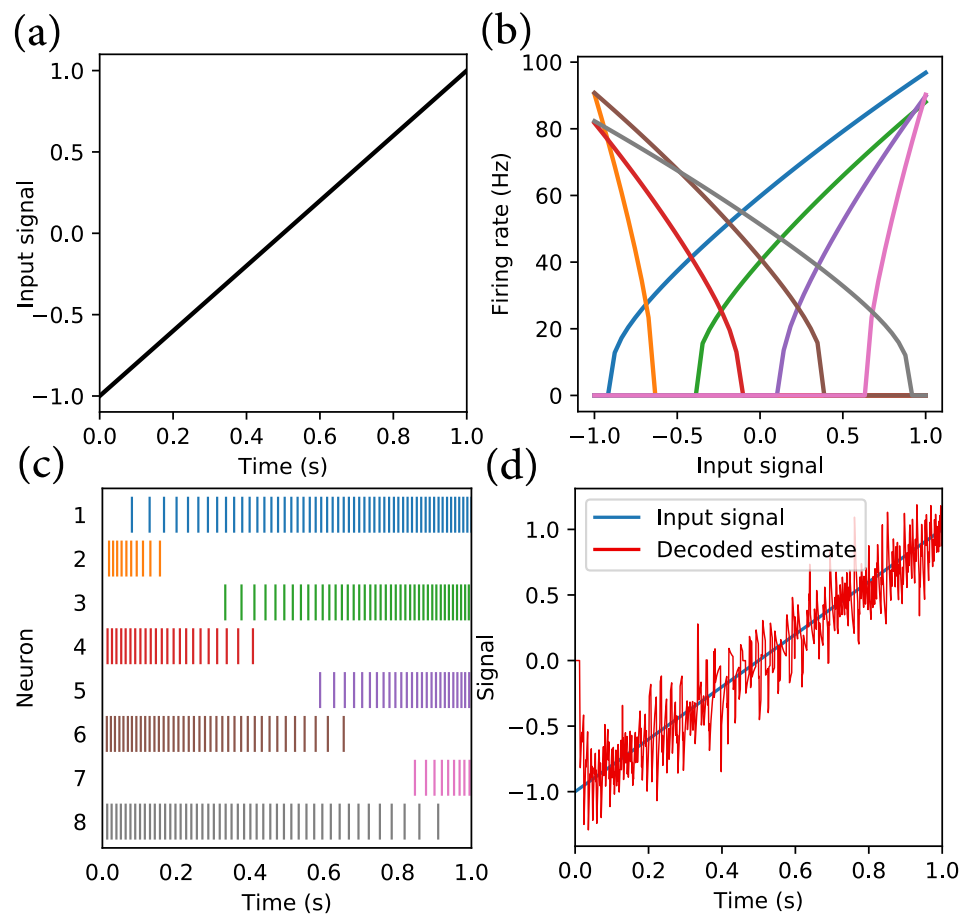


Figure 1. Illustration of encoding and decoding of signals using NEF. (a) normalised input signal, (b) tuning curves of 8 neurons, (c) spiking activity of neurons, (d) decoded signal compared with encoded signal [48].

2.1.2. Transformation

Arbitrary transformations of the signal can be decoded instead of the signal itself by simply solving for a different set of decoders \mathbf{d} . For a sample signal X , Equation (3) becomes

$$\mathbf{A}\mathbf{d} = f(\mathbf{X}) \quad (4)$$

where $f(\cdot)$ is the desired transform (when $f(\cdot)$ is known, decoders can thus be solved for. When only the output is known and the transformation is unknown, learning is involved).

2.1.3. Dynamics

An arbitrary dynamical system can be simulated using a network of spiking neurons where the state vectors of the dynamical system are represented by ensembles of neurons. Recurrent connections of ensembles allow differential equations to be simulated. The work presented in this paper does not involve simulating the dynamics of a system and hence does not utilise the third principle.

2.2. Cepstrum

The cepstrum was proposed by Bogert et al. in 1963 [49] and later developed by Oppenheim and Schafer [25]. Interpretively, it can be considered as the “spectrum” of the logarithm of a spectrum. The (power) cepstrum is usually given by:

$$\mathcal{C}(x(t)) = \left| \mathcal{F}^{-1} \left(\log \left(|\mathcal{F}(x(t))|^2 \right) \right) \right|^2 \quad (5)$$

where $\mathcal{F}(x)$ denotes the discrete Fourier transform (DFT). In practice, the DFT is performed using the fast Fourier transform (FFT) algorithm, and the magnitude squaring at the end may be omitted. The echo delay or period of a periodic signal τ appears as a peak in the cepstrum reminiscent of frequency peaks in conventional Fourier analysis. The domain of the cepstrum is thus termed as the quefrency domain. The cepstrum is particularly sensitive to reverberations or echoes of a fundamental wavelet, whose form does not have to be known a priori. This allows it to be particularly useful in the analysis of seismic signals and human speech, where it was initially used, and, later, in condition monitoring of rotating machinery. Health monitoring of infrastructure can also benefit from the cepstrum as it can eliminate harmonics and deconvolve periodic/quasi-periodic signals from the waveform [20,31–34].

Since our objective is recognition, rather than reconstruction, of the signal, a compressed version of the cepstrum can be generated by substituting the inverse DFT with the inverse discrete cosine transform (DCT). DCT improves compression due to its reduced complexity, and comparisons exist in the classic literature [50]. Analogous to Fourier components, a set of cepstral coefficients can thus be extracted, capturing the most relevant information in the signal. Damage to the structure would manifest as a change in cepstral coefficients which can be analysed by averaging over time. Deviation from the baseline healthy condition can be evaluated using simple statistical measures such as the Mahalanobis distance. This approach to SHM has been proposed previously with variations of cepstral features, with good results [31,51].

The process of extracting the cepstrum, i.e., decomposition into the log-spectrum and compression into coefficients centred at a finite number of characteristic frequencies, has been shown to be very similar to the process of feature extraction in the human ear [39]. A variant of the cepstrum called Mel-frequency cepstrum, where the frequency bands are spaced based on the human auditory response, has been widely used in speech processing. Passive and active mechanical components in the ear, along with neurons, extract the feature vector from the incoming sound and encode it in spikes sent to the auditory cortex of the brain. Considering the energy efficiency of the process, which is of critical importance as described earlier, this paper looks at a neuromorphic approach to extracting cepstral coefficients analogous to Mel-frequency cepstral coefficients (MFCCs) but distinct in the extraction procedure. The approach was first described by Bekolay [39] for the purpose of speech recognition and synthesis. While lack of powerful SNN training algorithms has posed challenges in terms of their performance as compared to artificial neural networks, recent efforts [52,53] have established the potential advantages of SNN in the context of MFCC by retaining numerical efficiency along with low-energy aspects.

3. Methods

3.1. Overview

The objective is to create a neuromorphic implementation of the cepstrum and test the feasibility of using it for SHM. For this, simulated and experimental datasets of a one-degree of freedom (1-DoF) oscillator with a change in stiffness were used. Once validated theoretically, the algorithm was implemented on hardware. The “Loihi” chip developed by Intel[®] was used to test the algorithm described in this paper. Loihi is a novel computing architecture composed of multiple cores composed of compartments of spiking neurons interconnected by synapses, allowing a direct mapping of a spiking neural network onto silicon [40]. The chip allows configuration of the network parameters such as synaptic delays as a hardware feature rather than emulated through algorithms. This allows SNNs to be flexibly implemented on the chip with minimal overhead, allowing high energy efficiency. The energy consumption per neuron update as well as synaptic operation (between neurons) is in the pico-Joule range on the chip, allowing it to achieve improvements in energy consumption of several orders of magnitude [54,55]. The SNN model was created using Nengo, a software library based on the NEF allowing an intuitive design environment [48], and deployed on the Loihi board using the Nengo Loihi backend.

3.2. Datasets

The simulated dataset was produced by the excitation of a single-degree-of-freedom linear oscillator (Figure 2a) using Gaussian white noise forcing for 60 s. The acceleration of the oscillator was measured. After recording the time series of the baseline undamaged state, the model was excited by the same force vector but with a sudden stiffness change $k' = 0.5 k$ introduced at 30 s. The same procedure was then repeated for a bilinear oscillator (Figure 2b), where α is changed from 1 to 0.5 at 30 s.

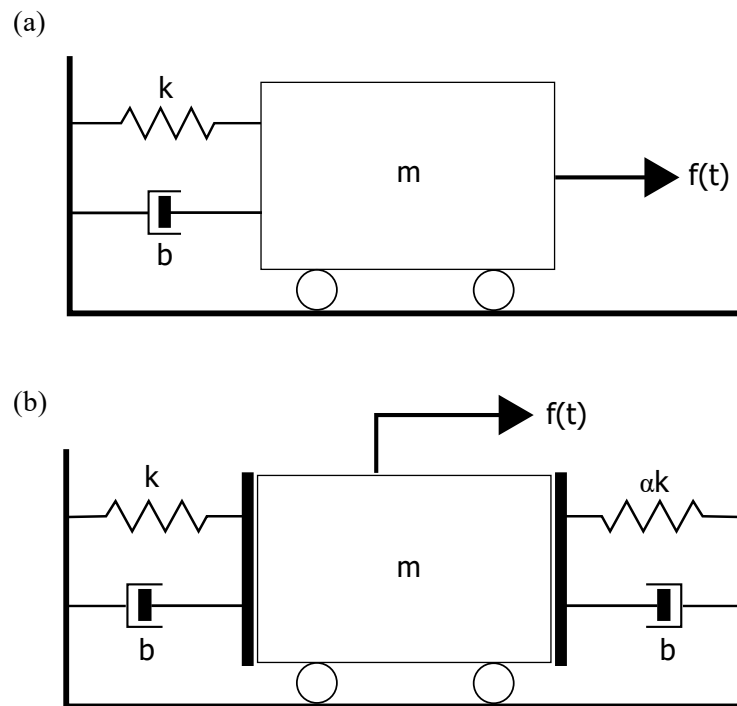


Figure 2. Schematic of simulated oscillator scenarios. (a) Linear and (b) bilinear 1-DoF systems were excited by a Gaussian white noise. Damage was introduced in the linear case by changing the stiffness k to $0.5 k$. For the bilinear case, damage was introduced by changing α from 1 to 0.5.

The experimental dataset was produced by the experimental setup shown in Figure 3. A single-degree-of-freedom cart coupled to the frame by 6 springs was excited using white noise. To simulate damage, springs were removed from the cart. To simulate robustness against changes in surface morphology, the experiment was conducted on wood, sandpaper, and plastic surfaces. The acceleration was measured using an accelerometer attached to the cart sampled at a rate of 617 data points per second. A detailed description of the experiment can be found in [56].

3.3. Architecture

The cepstrum may be considered as the composition of three functions:

- $\mathcal{F}(\cdot)$: Fourier transform;
- $2 \log(\cdot)$: Log transform;
- $\mathcal{F}^{-1}(\cdot)$: Inverse Fourier transform.

The final squaring operation may, in general, be omitted. The approach presented here is to use approximate methods to implement the transforms in a manner reminiscent of the human ear.

Passing the signals through a filterbank with different characteristic frequencies replicates the Fourier transform, as the output level of the filters provide an approximate frequency domain representation of the signal. Thus, a filterbank with logarithmically spaced characteristic frequencies would approximate the first two functions of the cepstrum, as listed above. For the third operation, the inverse discrete cosine transform (DCT-III

or iDCT) may be used in place of the inverse Fourier transform as it generates an almost uncorrelated representation while compressing data dimensionality. More importantly, it can be very efficiently implemented on hardware compared to an FFT because it is limited to the real domain.

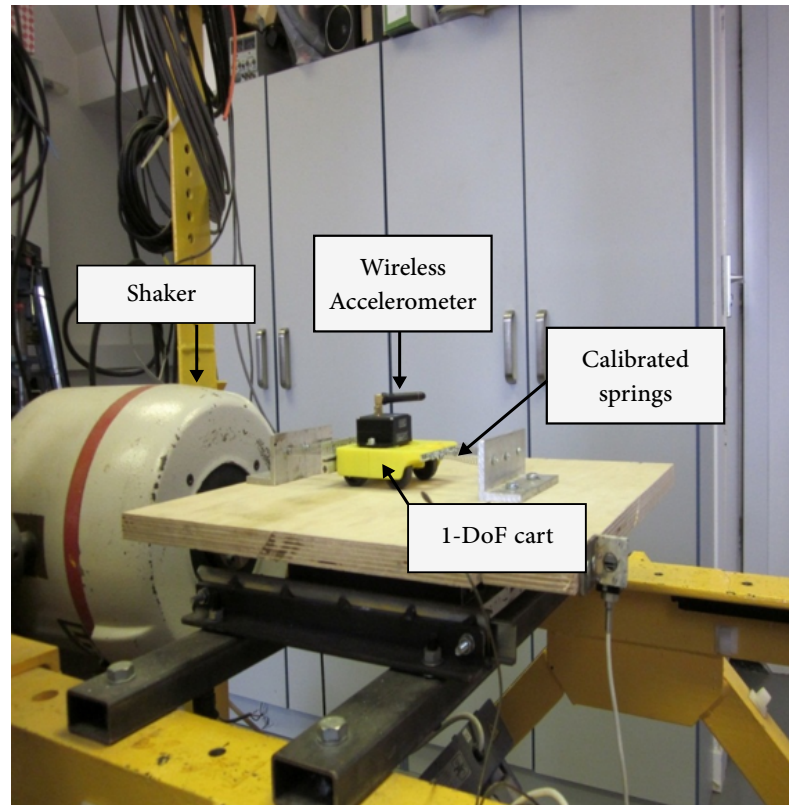


Figure 3. Setup for experimental test. The base on which the cart is placed was excited by Gaussian white noise by the shaker shown to the left. A total of 6 calibrated springs (3 on each side) are attached to the cart. The stiffness was changed in the damaged condition by removing 1 spring from each side.

3.4. Filterbank

As indicated earlier, cepstral analysis of speech/audio signals often uses the Mel-frequency scale and triangular filters. The Mel-scale is an empirically derived range approximating the human auditory range from 20–20,000 Hz. For SHM, a more compact range suffices, as most structures have modal frequencies in the sub 2000 Hz range. Instead of triangular filters, the gammatone filter is used, which is among the most widely used auditory filters [57]. The gammatone filter can be implemented very efficiently and can process signals in near-real time [58]. The impulse response of the filter is given by:

$$g(t) = at^{n-1}e^{-2\pi bt\text{ERB}(f)} \cos(2\pi ft), \quad (6)$$

$$\text{ERB}(f) = 24.7 + 0.108f \quad (7)$$

where a is the amplitude, n is filter order, b is the filter bandwidth parameter, and $\text{ERB}(f)$ is the equivalent rectangular bandwidth of the filter centred at frequency f . A filterbank of 36 filters logarithmically spaced in the 0–2000 Hz range was used (Figure 4). Increasing the number of filters beyond this did not appear to affect the results perceptibly.

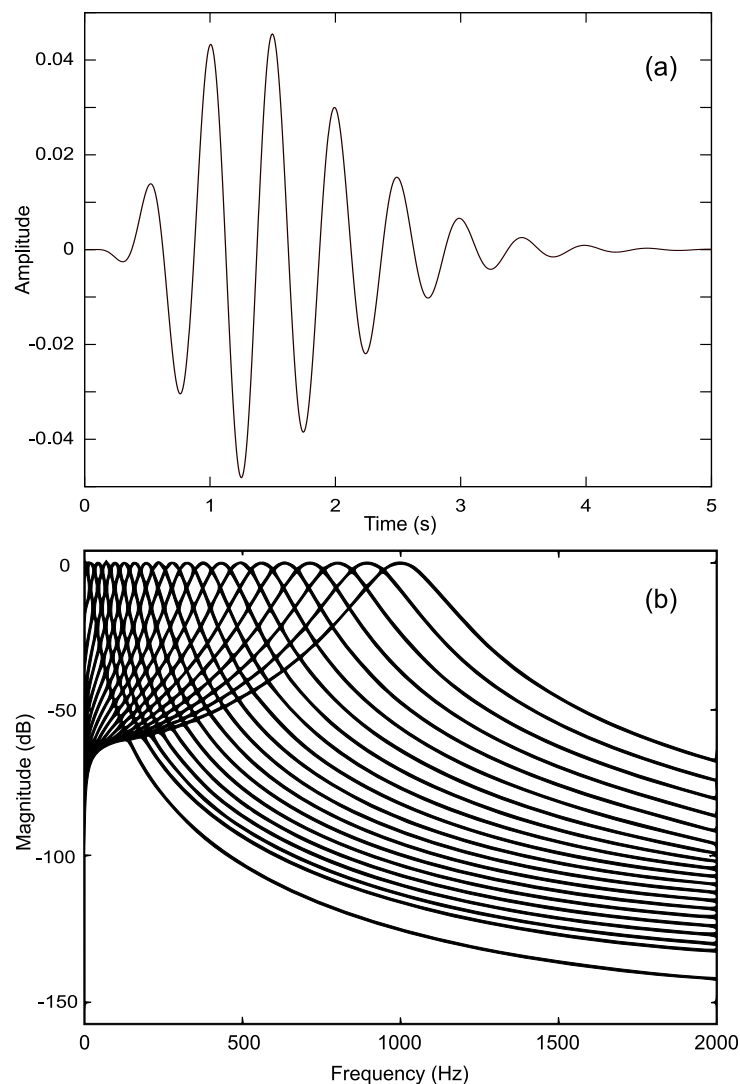


Figure 4. (a) Impulse response of gammatone filter. (b) Frequency response of filterbank (20/36 filters shown).

3.5. SNN

The SNN involves 2 layers of neurons. Signals from the m -dimensional filterbank are encoded into spiking signals in the first layer (see Figure 5). Leaky-integrate-and-fire (LIF) type neurons are used to encode the information. Twelve neurons are used per characteristic frequency. Intercepts are uniformly distributed in the range $(-0.1, 0.5)$ (the point along each neuron's encoder where its activity starts). The connection between the first and second layer implements the inverse DCT. The second layer thus gives the cepstral coefficients which compose the feature vector. The dimensionality of the feature vector is decided using the energy contained in the cepstral coefficients estimated using the L2-norm. The number of coefficients from the iDCT in the feature vector is incremented by 1 until $>99\%$ of the energy of the original signal is contained in the vector. To represent each cepstral coefficient, an ensemble of 20 neurons was used.

The minimum number of cepstral coefficients required for the datasets in this study was found to be 6. Thus, for encoding the signal using 36 filterbanks, a total of $36 \times 12 = 432$ neurons were used. To decode the 6 iDCT coefficients, $6 \times 20 = 120$ neurons were used. The spiking neural network thus required 552 neurons to implement.

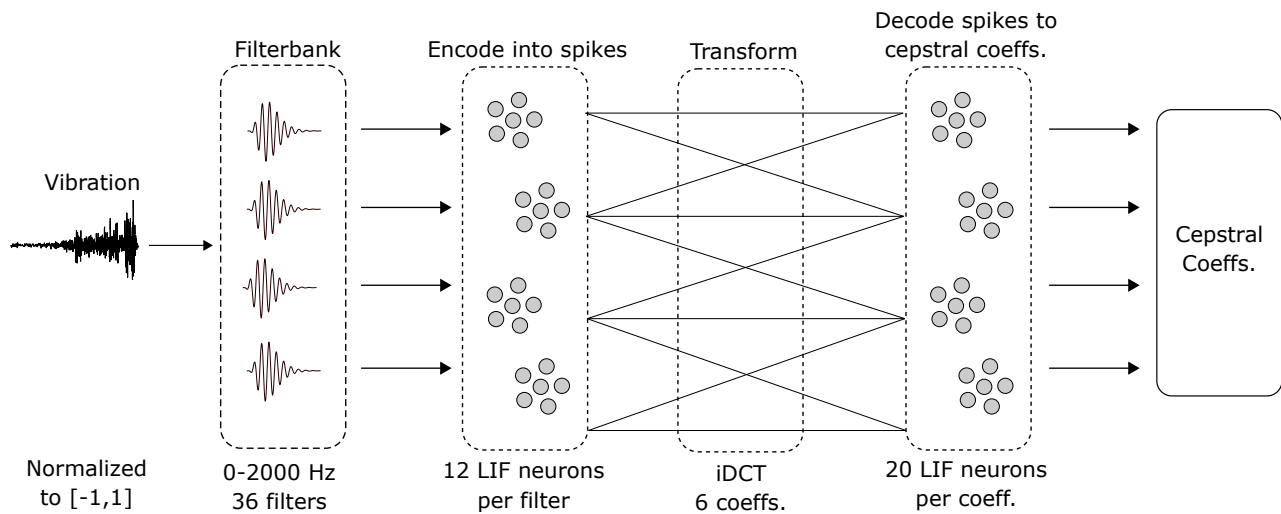


Figure 5. Schematic of the SNN architecture. The input signal is passed through filters with logarithmically spaced characteristic frequencies and encoded into spikes. An ensemble of spiking neurons is allocated to each filter. The inverse DCT transformation is performed on the output of the first layer and passed to the second layer. The transformation is performed through weights on the connections between the layers. The cepstral coefficients are decoded from the second layer. The number of coefficients to be extracted is chosen such that the L2-norm of all the coefficients combined captures >99% of the energy of the input signal. The number of ensembles is the same as the number of cepstral coefficients. The architecture is adapted from the Sermo model proposed by Bekolay [39].

3.6. Damage Classification

The damaged state is identified using the Mahalanobis distance of the feature vector averaged in time from the distribution of the feature vector of the undamaged state. The Mahalanobis distance of vector \mathbf{x} from the baseline multivariate distribution with mean vector $\boldsymbol{\mu}$ is measured as

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^T S^{-1} (\mathbf{x} - \boldsymbol{\mu})} \quad (8)$$

where S is the covariance matrix of the baseline distribution. An averaging window of 1.5 s is used. Threshold identification of the undamaged vs. damaged condition can be carried out based on tolerance levels required, as shown in Balsamo et al. [31].

4. Results

The results for the simulated case (Figure 6) show a clear change in the cepstral coefficients after $t = 30$ s, for both the linear and bilinear case, after which the Mahalanobis distance of the damaged state remains higher than the undamaged state overall. The oscillations in the Mahalanobis distance are smoothed if a higher averaging time (currently 1.5 s) is used, allowing a simple classification of damaged and undamaged states at very short timescales.

For the experimental case, the baseline is the undamaged oscillation over a wood surface. Figure 7a–c show the Mahalanobis distances from the mean of the undamaged condition for sandpaper, plastic, and wood, respectively. All three plots show a clear differentiation in the feature vectors between damaged and undamaged, allowing a simple classification into damaged and undamaged states. The damaged states are always at a higher distance from the undamaged state. Figure 7d shows a comparison between undamaged conditions for two surfaces (wood and plastic), showing that the feature is robust to variations in surface conditions.

The static and dynamic power consumption of the chip were measured (using profiling tools provided through the Loihi SDK) to be approximately 1.0 Watt and 0.1 Watt, respectively.

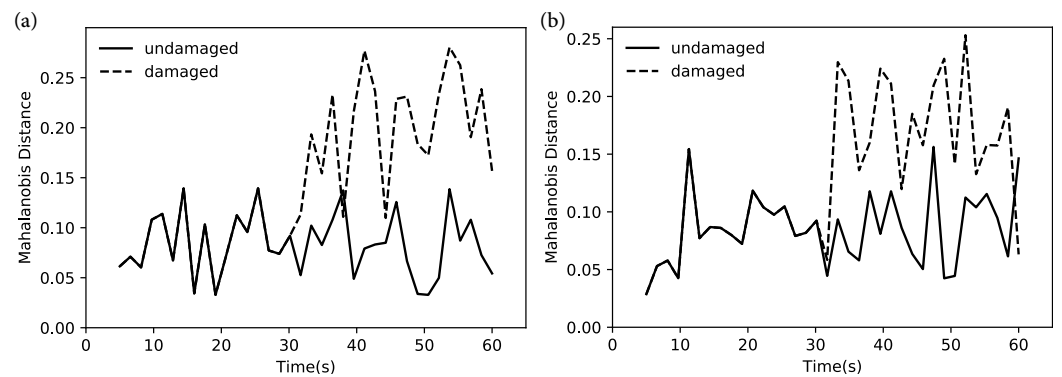


Figure 6. Change in Mahalanobis distance between the feature vectors of the damaged and undamaged states. (a) Simulated linear system and (b) simulated bilinear system. Damage introduced at $t = 30$ s.

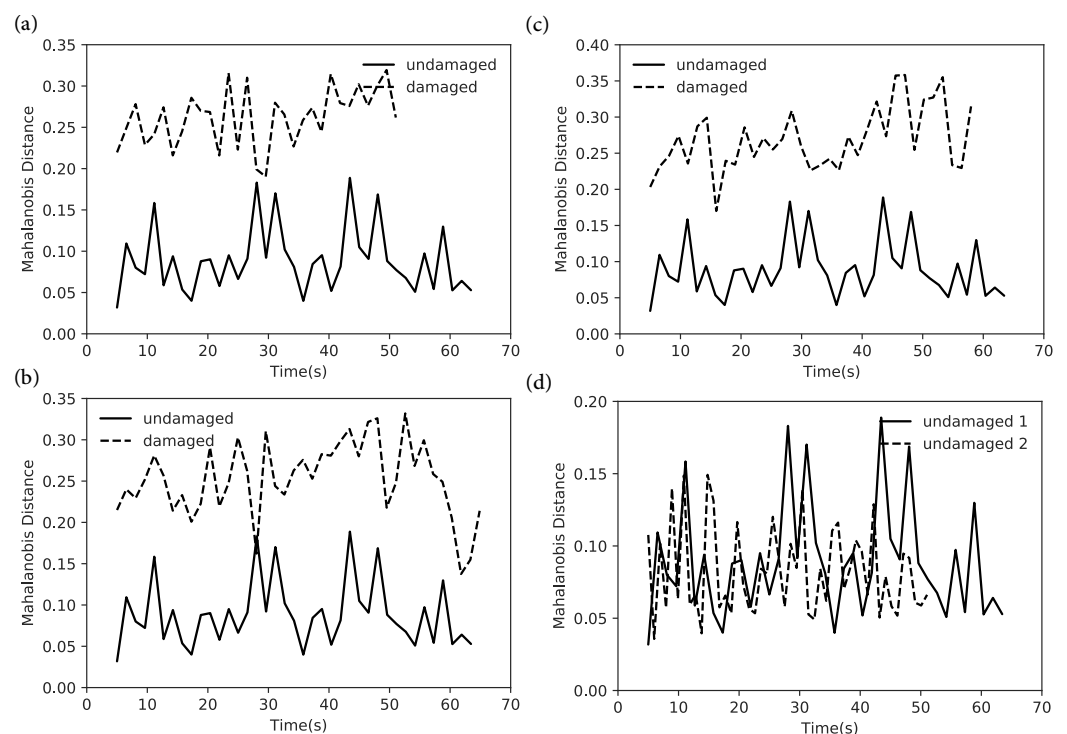


Figure 7. Change in Mahalanobis distance between the feature vectors of the damaged and undamaged states in experimental cases with surfaces: (a) Sandpaper, (b) plastic, and (c) wood. Mahalanobis distance comparison of the undamaged cases for wood and plastic is shown in (d).

5. Discussion and Conclusions

An SNN-based implementation is not expected to be more efficient than an optimised hardware implementation of the MFCC algorithm. However, the conceptual translation of a well-defined and well-understood transform to the spiking domain highlights the potential to use spiking neural networks as a programmable and explainable paradigm for feature extraction in SHM while also offering benefits of low-power operation. The scalability of neural networks to arbitrary complexity in feature extraction would also allow highly sophisticated pattern recognition from sensor streams across many degrees of freedom of the structure, or to enable sensor fusion from multimodal, multiresolution sensors.

Loihi contains 128 neuromorphic cores, each with 1024 spiking compartments. The implementation presented here utilises less than 1% of the available compartments on the chip. A fully utilised chip consumes less than 2 Watts, implying that multiple streams of sensor input could be processed in near-real time on a single chip to extract damage-

sensitive features at a node. Alternatively, more advanced neuromorphic algorithms may be implemented at the node level to extract better damage-sensitive features to be transmitted.

Static power dominates the power consumption in Loihi at ~ 1 W. However, an application-specific integrated circuit (ASIC) for SHM could be envisaged with a subset of neurons and optimised functions for pre-/post-processing that brings the resource utilisation closer to 100% operating at the mW range or lower. Several digital and mixed-signal implementations of spiking neural network accelerators have been developed recently that consume power as low as 100 μ W [59–63]. The prospects that these architectures offer are perfectly suited for SHM since true edge deployment of sophisticated damage-detection algorithms may become viable once a sufficiently low threshold of system power (including acquisition and transmission) is crossed.

The damage identification using Mahalanobis distance was carried out using feature vectors from a single sensor. An averaging window of just 1.5 s results in a clear distinction in the damaged and undamaged state, as shown. This confirms that the SNN-based cepstrum generates damage-sensitive features. It is reasonable to project that integration of data from multiple sensors over larger windows of time along with pattern recognition algorithms suited for neuromorphic implementation would allow accurate damage identification at very low power levels.

The novelty introduced in this work is the use of a neuromorphic architecture (non-Von Neumann) as opposed to a conventional processing architecture (such as x86). The cepstrum, by virtue of its biological similarity, lends itself to a simple adaptation. However, arbitrary transformations of the input signal can be implemented using the methods as described in the NEF. Algorithms that rely on incorporate online learning rather than a pre-determined explicit transform would gain higher advantages from an SNN implementation. The prospects offered by such advanced processing capabilities at a node with low power demand open avenues for SHM implementation that were not feasible previously.

Author Contributions: Conceptualisation, G.V.J. and V.P.; methodology, G.V.J. and V.P.; software, G.V.J.; validation, G.V.J. and V.P.; formal analysis, G.V.J.; investigation, G.V.J. and V.P.; resources, V.P.; data curation, V.P.; writing—original draft preparation, G.V.J.; writing—review and editing, V.P.; visualisation, G.V.J. and V.P.; supervision, V.P.; project administration, V.P.; funding acquisition, V.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received funding from Science Foundation Ireland MaREI project RC2302 2 Accenture NeuroSHM project Science Foundation Ireland NexSys 21/SPP/3756 Enterprise Ireland SEMPRES project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data are available upon reasonable request.

Acknowledgments: The authors would like to acknowledge the Intel Neuromorphic Research Community.

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

Appendix A

The Neural Engineering Framework (NEF) allows arbitrary dynamical systems to be described using spiking or nonspiking neural networks. A formal description of the framework is presented below. The NEF can approximate exact mathematical operations with precision of $O(\sqrt{n})$ for spiking networks and $O(n)$ for the nonspiking case. The framework can be summarised using three principles: representation, transformation, and dynamics.

Principle 1: Representation

The scheme of distributed information representation in the NEF is called population coding, where an ensemble of neurons collectively encode information. The scheme was

proposed in 1986 based on empirical evidence from the monkey cortex. A spiking neuron is a biologically derived model where the neuron “fires” only when the membrane potential, an intrinsic property of the neuron, crosses a threshold. It is also called the integrate-and-fire model. We represent the model using a capacitance C :

$$\frac{dV(t)}{dt} = \frac{1}{C}J(t) \quad (\text{A1})$$

where the voltage $V(t)$ increases with time until it reaches a threshold, when a spike is emitted and the voltage is reset. A biological implausibility in this model is that it retains any energy input indefinitely. To overcome this, a leaky integrate-and-fire (LIF) neuron model was proposed, which can be represented as:

$$\frac{dV(t)}{dt} = \frac{1}{RC}(J(t)R - V(t)) \quad (\text{A2})$$

Incorporating a refractory period after each spike τ_{ref} , during which spiking cannot happen for a constant input (thereby limiting spike frequency), the activity of a neuron can be solved exactly as

$$s(J) = \begin{cases} \frac{1}{\tau_{\text{ref}} - \tau_{\text{RC}} \ln(1 - \frac{J_{\text{th}}}{J})} & \text{if } J > J_{\text{th}} \\ 0 & \text{Otherwise} \end{cases} \quad (\text{A3})$$

where J is the input current, J_{th} is the threshold at which spiking occurs, and τ_{RC} is the time constant for the rate of charge accumulation.

Using the LIF neuron model, the NEF describes a nonlinear encoding and linear decoding scheme to represent a q -dimensional vector in a population of n spiking neurons. Each neuron’s activity a is a function of the current J injected into it, e.g., spiking begins in a neuron at a current threshold and the spike rate increases with current. A different neuron may have the inverse behaviour, where spiking decreases with current and stops at a current threshold. An ensemble of n -neurons of both “positive” and “negative” spiking directions as $n \rightarrow \infty$ can thus exactly represent arbitrary signals using thresholds at all real numbers. The input current vector to encode the signal vector x is thus:

$$\mathbf{J}(\mathbf{x}) = \alpha \mathbf{e} \cdot \mathbf{x} + \mathbf{J}_{\text{bias}} \quad (\text{A4})$$

where \mathbf{J}_{bias} determines the J_{th} (intercept/threshold) of each neuron, \mathbf{e} gives the direction, and α gives the rate of increase of the spiking frequency with signal amplitude. In practice, a finite domain of $[-1, 1]$ is used, scaling the signals to the domain, and outliers are effectively clipped at the boundaries. Intercepts are typically distributed equally between positive and negative directions sampled uniformly (see Figure 1). However, to optimise the usage of each neuron, the choice of intercepts may be skewed or shifted based on the system modelled, e.g., if only the threshold-crossing events of a signal in the positive direction, say ≥ 0.9 , are to be encoded, all encoders \mathbf{e} may be set to be positive and all elements of \mathbf{J}_{bias} may be set to 0.9. The activity of the ensemble using Equation (A3) is thus $\mathbf{a}(\mathbf{J})$, where the transfer function of a single neuron, as shown in Equation (A3), is used, giving

$$\mathbf{a}(\mathbf{J}) = \mathbf{s}(\mathbf{J}) \quad (\text{A5})$$

Decoding encoded signals from the activity $\mathbf{a}(\mathbf{J})$ amounts to estimating the vector \mathbf{x} as the weighted sum of each neuron’s activity. The weights are called decoders in the NEF.

$$\hat{\mathbf{x}} = \sum_{i=1}^n \mathbf{d}_i a_i \quad (\text{A6})$$

where $\hat{\mathbf{x}}$ is the decoded estimate of \mathbf{x} , \mathbf{d}_i is the set of decoding weights for each neuron, and a_i is the spiking activity of the neuron. Since the characteristics of each neuron are known,

the activity generated by all possible inputs is also known. This allows the decoding weights to be solved for by setting up the equation

$$\mathbf{A}\mathbf{d} = \mathbf{X} \quad (\text{A7})$$

where \mathbf{X} is set of sample inputs and \mathbf{A} is the set of activities of all neurons n for all possible inputs m , given by

$$\mathbf{A} = a_j(X_i) \quad (\text{A8})$$

where $i = 1, 2, \dots, m$ and $j = 1, 2, \dots, n$. An approximation of the decoders \mathbf{d} is obtained by solving the least squares problem given by Equation (A7).

Since the spikes generated by the neurons take a finite amount of time to decay, a post-synaptic filter $h(t)$ is introduced to the decoding operation. A basic model for the filter is as a decaying exponential:

$$h(t) = \frac{1}{\tau_{\text{PSC}}} e^{-t/\tau_{\text{PSC}}} \quad (\text{A9})$$

Thus, the activity of a neuron is

$$a(t) = \sum_s \delta(t - t_s) * h(t) = \sum_s h(t - t_s) \quad (\text{A10})$$

where δ is the Dirac delta function for the spikes, s is the set of spike times, and $*$ represents convolution.

The decoded estimate $\hat{\mathbf{x}}$ is then:

$$\hat{\mathbf{x}} = \sum_{i=1}^n \sum_s h(t - t_{i,s}) \mathbf{d}_i \quad (\text{A11})$$

Principle 2: Transformation

The connections between neurons allow transformations of the encoded signal before decoding using weights on the connections. The activity of n neurons from an ensemble acts as the input current to another neuron j in an ensemble:

$$J_j(x) = \sum_{i=1}^n \omega_{ij} a_i + J_{j,\text{bias}} \quad (\text{A12})$$

where ω_{ij} is the connection strength between neurons i and j . If no transformation is applied on the signal, using Equations (A4) and (A6), the weights can be expressed as

$$\omega_{ij} = \alpha_j \mathbf{e}_j \cdot \mathbf{d}_i \quad (\text{A13})$$

If a linear transformation is to be implemented on the signal, the input current to the downstream ensemble can be modified by changing ω_{ij} as

$$\omega_{ij} = \alpha_j \mathbf{e}_j \mathbf{L}_{ji} \mathbf{d}_i \quad (\text{A14})$$

where L_{ji} gives the transformation matrix. Thus, for linear transformation, the decoders are simply scaled using the transformation matrix to obtain the transformed output from the network.

For an arbitrary function $f(\mathbf{x})$, the computation of weights, and consequently decoders, becomes an optimisation problem, as shown below.

The estimate of the output to be obtained after decoding is

$$(f(\mathbf{x}) * h)(t) \approx \sum_{i=1}^n (a_i * h)(t) \mathbf{d}_i \quad (\text{A15})$$

$$= \sum_{i=1}^n \sum_s h(t - t_{i,s}) \mathbf{d}_i \quad (\text{A16})$$

The activity of a single neuron for constant input is given by Equation (A3). Introducing a white noise process η to account for the variance of spiking across the population, the solution to the set of decoders d in Equation (A16) is given by the minimisation problem:

$$\mathbf{d} = \underset{\mathbb{R}^{n \times q}}{\operatorname{argmin}} \int_m \left\| f(\mathbf{v}) - \sum_{i=1}^n (s_i(\mathbf{v} + \eta)) \mathbf{d}_i \right\|_2^2 d\mathbf{v} \quad (\text{A17})$$

where m is the domain of the q -dimensional signal \mathbf{x} and $\mathbf{v} \in m$. The optimisation does not depend on the nature of x , but only on the distribution of the signal. The dynamic problem involving spiking has been converted to a static problem, allowing the decoders to be computed without explicitly simulating the neurons in time. This convex optimisation problem can be solved by sampling the domain m uniformly and solving the resulting least-squares problem, similar to Equation (A7).

Principle 3: Dynamics

A linear time-invariant (LTI) system can be represented as

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) \quad (\text{A18})$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t) \quad (\text{A19})$$

The ability to simulate dynamical systems relies on the availability of an integrator, i.e., the source of dynamics is the integration over time of the system above. In the NEF, the integrator is replaced by a *leaky integrator* given by the low-pass filter $h(t)$ available on the connections between the neurons.

$$h(t) = \frac{1}{\tau} e^{-t/\tau} = \mathcal{L}^{-1} \left(\frac{1}{1 + \tau s} \right) \quad (\text{A20})$$

where $\mathcal{L}^{-1}(\cdot)$ is the inverse Laplace transform.

Thus, a recurrent connection in an ensemble allows the integration of Equation (A19) by replacing A and B as follows:

$$\mathbf{A}' = \tau \mathbf{A} + \mathbf{I} \quad (\text{A21})$$

$$\mathbf{B}' = \tau \mathbf{B} \quad (\text{A22})$$

where \mathbf{I} is the identity matrix. In other words, the replacement of the integrator by the filter is compensated by driving the synapse with $\tau \dot{\mathbf{x}}(t) + \mathbf{x}(t)$. This transformation of $\mathbf{x}(t)$ may be implemented using Principle 2, thereby allowing simulation of dynamical systems.

References

1. Farrar, C.R.; Worden, K. An introduction to structural health monitoring. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2007**, *365*, 303–315. [CrossRef]
2. Ko, J.; Ni, Y. Technology developments in structural health monitoring of large-scale bridges. *Eng. Struct.* **2005**, *27*, 1715–1725. [CrossRef]
3. Kim, S.; Pakzad, S.; Culler, D.; Demmel, J.; Fenves, G.; Glaser, S.; Turon, M. Health monitoring of civil infrastructures using wireless sensor networks. In Proceedings of the IPSN 2007: The Sixth International Symposium on Information Processing in Sensor Networks, Cambridge, MA, USA, 25–27 April 2007; ACM Press: New York, NY, USA, 2007; pp. 254–263. [CrossRef]

4. Lynch, J.P.; Loh, K.J. A Summary Review of Wireless Sensors and Sensor Networks for Structural Health Monitoring. *Shock Vib. Dig.* **2006**, *38*, 91–128. [[CrossRef](#)]
5. Spencer, B.F.; Ruiz-Sandoval, M.E.; Kurata, N. Smart sensing technology: Opportunities and challenges. *Struct. Control. Health Monit.* **2004**, *11*, 349–368. [[CrossRef](#)]
6. Cao, S.; Li, J. A survey on ambient energy sources and harvesting methods for structural health monitoring applications. *Adv. Mech. Eng.* **2017**, *9*, 168781401769621. [[CrossRef](#)]
7. Tokogonon, A.C.; Gao, B.; Tian, G.Y.; Yan, Y. Structural Health Monitoring Framework Based on Internet of Things: A Survey. *IEEE Internet Things J.* **2017**, *4*, 629–635. [[CrossRef](#)]
8. Alonso, L.; Barbaran, J.; Chen, J.; Diaz, M.; Llopis, L.; Rubio, B. Middleware and communication technologies for structural health monitoring of critical infrastructures: A survey. *Comput. Stand. Interfaces* **2018**, *56*, 83–100. [[CrossRef](#)]
9. Loubet, G.; Takacs, A.; Gardner, E.; De Luca, A.; Udea, F.; Dragomirescu, D. LoRaWAN Battery-Free Wireless Sensors Network Designed for Structural Health Monitoring in the Construction Domain. *Sensors* **2019**, *19*, 1510. [[CrossRef](#)] [[PubMed](#)]
10. Buckley, T.; Ghosh, B.; Pakrashi, V. A Feature Extraction & Selection Benchmark for Structural Health Monitoring. *Struct. Health Monit.* **2022**, 14759217221111141. [[CrossRef](#)]
11. Buckley, T.; Ghosh, B.; Pakrashi, V. Edge structural health monitoring (E-SHM) using low-power wireless sensing. *Sensors* **2021**, *21*, 6760. [[CrossRef](#)]
12. Abdaoui, A.; El Fouly, T.M.; Ahmed, M.H. Impact of time synchronization error on the mode-shape identification and damage detection/localization in WSNs for structural health monitoring. *J. Netw. Comput. Appl.* **2017**, *83*, 181–189. [[CrossRef](#)]
13. Krishnan, M.; Bhowmik, B.; Hazra, B.; Pakrashi, V. Real time damage detection using recursive principal components and time varying auto-regressive modeling. *Mech. Syst. Signal Process.* **2018**, *101*, 549–574. [[CrossRef](#)]
14. Srbinovski, B.; Magno, M.; O’Flynn, B.; Pakrashi, V.; Popovici, E. Energy aware adaptive sampling algorithm for energy harvesting wireless sensor networks. In Proceedings of the 2015 IEEE Sensors Applications Symposium (SAS), Zadar, Croatia, 13–15 April 2015; pp. 1–6. [[CrossRef](#)]
15. Vathakkattil Joseph, G.; Hao, G.; Pakrashi, V. Extreme value estimates using vibration energy harvesting. *J. Sound Vib.* **2018**, *437*, 29–39.
16. Vathakkattil Joseph, G.; Hao, G.; Pakrashi, V. Fragility analysis using vibration energy harvesters. *Eur. Phys. J. Spec. Top.* **2019**, *228*, 1625–1633. [[CrossRef](#)]
17. Farrar, C.; Worden, K. *Structural Health Monitoring: A Machine Learning Perspective*; John Wiley & Sons, Ltd.: Hoboken, NJ, USA, 2012.
18. Min, J.; Park, S.; Yun, C.B.; Lee, C.G.; Lee, C. Impedance-based structural health monitoring incorporating neural network technique for identification of damage type and severity. *Eng. Struct.* **2012**, *39*, 210–220. [[CrossRef](#)]
19. Bandara, R.P.; Chan, T.H.; Thambiratnam, D.P. Structural damage detection method using frequency response functions. *Struct. Health Monit.* **2014**, *13*, 418–429. [[CrossRef](#)]
20. Dackermann, U.; Smith, W.A.; Randall, R.B. Damage identification based on response-only measurements using cepstrum analysis and artificial neural networks. *Struct. Health Monit.* **2014**, *13*, 430–444. [[CrossRef](#)]
21. Guo, T.; Wu, L.; Wang, C.; Xu, Z. Damage detection in a novel deep-learning framework: A robust method for feature extraction. *Struct. Health Monit.* **2020**, *19*, 424–442. [[CrossRef](#)]
22. Mandal, S.; Zhak, S.M.; Sarpeshkar, R. A bio-inspired active radio-frequency silicon cochlea. *IEEE J. Solid-State Circuits* **2009**, *44*, 1814–1828. [[CrossRef](#)]
23. Lobo, J.L.; Del Ser, J.; Bifet, A.; Kasabov, N. Spiking Neural Networks and online learning: An overview and perspectives. *Neural Netw.* **2020**, *121*, 88–100.
24. Nawrocki, R.A.; Voyles, R.M.; Shaheen, S.E. A Mini Review of Neuromorphic Architectures and Implementations. *IEEE Trans. Electron Devices* **2016**, *63*, 3819–3829. [[CrossRef](#)]
25. Oppenheim, A.; Schaffer, R. From frequency to quefrequency: A history of the cepstrum. *IEEE Signal Process. Mag.* **2004**, *21*, 95–106. [[CrossRef](#)]
26. Childers, D.G.; Skinner, D.P.; Kemerait, R.C. The Cepstrum: A Guide to Processing. *Proc. IEEE* **1977**, *65*, 1428–1443. [[CrossRef](#)]
27. Henriquez, P.; Alonso, J.B.; Ferrer, M.A.; Travieso, C.M. Review of automatic fault diagnosis systems using audio and vibration signals. *Systems* **2014**, *44*, 642–652. [[CrossRef](#)]
28. Liang, B.; Iwnicki, S.D.; Zhao, Y. Application of power spectrum, cepstrum, higher order spectrum and neural network analyses for induction motor fault diagnosis. *Mech. Syst. Signal Process.* **2013**, *39*, 342–360. [[CrossRef](#)]
29. Fazel, A.; Chakraborty, S. An overview of statistical pattern recognition techniques for speaker verification. *IEEE Circuits Syst. Mag.* **2011**, *11*, 62–81. [[CrossRef](#)]
30. Zheng, F.; Zhang, G.; Song, Z. Comparison of different implementations of MFCC. *J. Comput. Sci. Technol.* **2001**, *16*, 582–589. [[CrossRef](#)]
31. Balsamo, L.; Betti, R.; Beigi, H. A structural health monitoring strategy using cepstral features. *J. Sound Vib.* **2014**, *333*, 4526–4542. [[CrossRef](#)]
32. Mei, Q.; Gul, M. A crowdsourcing-based methodology using smartphones for bridge health monitoring. *Struct. Health Monit.* **2019**, *18*, 1602–1619. [[CrossRef](#)]
33. Dackermann, U.; Smith, W.A.; Alamdari, M.M.; Li, J.; Randall, R.B. Cepstrum-based damage identification in structures with progressive damage. *Struct. Health Monit.* **2019**, *18*, 87–102. [[CrossRef](#)]

34. Cheng, H.; Wang, F.; Huo, L.; Song, G. Detection of sand deposition in pipeline using percussion, voice recognition, and support vector machine. *Struct. Health Monit.* **2020**, 147592172091889. [[CrossRef](#)]
35. Lakshmi, K.; Rao, A.R.M.; Gopalakrishnan, N. Singular spectrum analysis combined with ARMAX model for structural damage detection. *Struct. Control Health Monit.* **2017**, *24*, e1960. [[CrossRef](#)]
36. Cao, Y.; Chen, Y.; Khosla, D. Spiking Deep Convolutional Neural Networks for Energy-Efficient Object Recognition. *Int. J. Comput. Vis.* **2015**, *113*, 54–66. [[CrossRef](#)]
37. Esser, S.K.; Merolla, P.A.; Arthur, J.V.; Cassidy, A.S.; Appuswamy, R.; Andreopoulos, A.; Berg, D.J.; McKinstry, J.L.; Melano, T.; Barch, D.R.; et al. Convolutional networks for fast, energy-efficient neuromorphic computing. *Proc. Natl. Acad. Sci. USA* **2016**, *113*, 11441–11446.
38. Han, B.; Sengupta, A.; Roy, K. On the energy benefits of spiking deep neural networks: A case study. In Proceedings of the International Joint Conference on Neural Networks, Vancouver, BC, Canada, 24–29 July 2016; Institute of Electrical and Electronics Engineers Inc.: New York City, NY, USA, 2016; Volume 2016, pp. 971–976. [[CrossRef](#)]
39. Bekolay, T. Biologically Inspired Methods in Speech Recognition and Synthesis: Closing The Loop. Ph.D. Thesis, University of Waterloo, Waterloo, ON, Canada, 2016.
40. Davies, M.; Srinivasa, N.; Lin, T.H.; China, G.; Cao, Y.; Choday, S.H.; Dimou, G.; Joshi, P.; Imam, N.; Jain, S.; et al. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* **2018**, *38*, 82–99. [[CrossRef](#)]
41. Pang, L.; Liu, J.; Harkin, J.; Martin, G.; McElholm, M.; Javed, A.; McDaid, L. Case Study—Spiking Neural Network Hardware System for Structural Health Monitoring. *Sensors* **2020**, *20*, 5126. [[CrossRef](#)]
42. Zanatta, L.; Barchi, F.; Burrello, A.; Bartolini, A.; Brunelli, D.; Acquaviva, A. Damage Detection in Structural Health Monitoring with Spiking Neural Networks. In Proceedings of the 2021 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4.0&IoT), Rome, Italy, 7–9 June 2021; pp. 105–110.
43. Micu, E.A.; O'Brien, E.J.; Bowe, C.; Fitzgerald, P.; Pakrashi, V. Bridge damage and repair detection using an instrumented train. *J. Bridge Eng.* **2022**, *27*, 05021018. [[CrossRef](#)]
44. Inturi, V.; Balaji, S.V.; Gyanam, P.; Pragada, B.P.V.; Geetha Rajasekharan, S.; Pakrashi, V. An integrated condition monitoring scheme for health state identification of a multi-stage gearbox through Hurst exponent estimates. *Struct. Health Monit.* **2022**, 14759217221092828. [[CrossRef](#)]
45. Eliasmith, C.; Anderson, C. *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*; MIT Press: Cambridge, MA, USA, 2004.
46. Voelker, A.R. Dynamical Systems in Spiking Neuromorphic Hardware. Ph.D. Thesis, University of Waterloo: Waterloo, ON, Canada, 2019.
47. Deng, L.; Wu, Y.; Hu, X.; Liang, L.; Ding, Y.; Li, G.; Zhao, G.; Li, P.; Xie, Y. Rethinking the performance comparison between SNNS and ANNS. *Neural Netw.* **2020**, *121*, 294–307. [[CrossRef](#)]
48. Bekolay, T.; Bergstra, J.; Hunsberger, E.; DeWolf, T.; Stewart, T.C.; Rasmussen, D.; Choo, X.; Voelker, A.R.; Eliasmith, C. Nengo: A Python tool for building large-scale functional brain models. *Front. Neuroinform.* **2014**, *7*, 48. [[CrossRef](#)]
49. Bogert, B.P. The quefrency analysis of time series for echoes; Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. *Time Ser. Anal.* **1963**, 209–243.
50. Vetterli, M.; Nussbaumer, H.J. Simple FFT and DCT algorithms with reduced number of operations. *Signal Process.* **1984**, *6*, 267–278. [[CrossRef](#)]
51. Civera, M.; Ferraris, M.; Ceravolo, R.; Surace, C.; Betti, R. The Teager-Kaiser Energy Cepstral Coefficients as an Effective Structural Health Monitoring Tool. *Appl. Sci.* **2019**, *9*, 5064. [[CrossRef](#)]
52. Martinelli, F.; Dellaferrera, G.; Mainar, P.; Cernak, M. Spiking neural networks trained with backpropagation for low power neuromorphic implementation of voice activity detection. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 8544–8548.
53. Wu, X.; Dang, B.; Wang, H.; Wu, X.; Yang, Y. Spike-Enabled Audio Learning in Multilevel Synaptic Memristor Array-Based Spiking Neural Network. *Adv. Intell. Syst.* **2022**, *4*, 2100151. [[CrossRef](#)]
54. Tang, G.; Shah, A.; Michmizos, K.P. Spiking Neural Network on Neuromorphic Hardware for Energy-Efficient Unidimensional SLAM. Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Macau, China, 3–8 November 2019; Institute of Electrical and Electronics Engineers Inc.: New York City, NY, USA, 2019; pp. 4176–4181.
55. Blouw, P.; Choo, X.; Hunsberger, E.; Eliasmith, C. Benchmarking keyword spotting efficiency on neuromorphic hardware. In Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop, Albany, NY, USA, 26–28 March 2019; ACM International Conference Proceeding Series; Association for Computing Machinery: New York, New York, USA, 2019; pp. 1–8.
56. Jaksic, V. Bridge-Vehicle Interaction for Structural Health Monitoring: Potentials, Applications, and Limitations. Ph.D. Thesis, University College Cork, Cork, Ireland, 2014.
57. Patterson, R.D. Auditory filter shapes derived with noise stimuli. *J. Acoust. Soc. Am.* **1976**, *59*, 640–654. [[CrossRef](#)]
58. Slaney, M. *An Efficient Implementation of the Patterson-Holdsworth Auditory Filter Bank*; Technical Report; Apple Computer, Inc.: Cupertino, CA, USA, 1993.
59. Frenkel, C.; Indiveri, G. ReckOn: A 28nm Sub-mm² Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales. In Proceedings of the 2022 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, USA, 20–26 February 2022; Volume 65, pp. 1–3.

60. Buhler, F.N.; Brown, P.; Li, J.; Chen, T.; Zhang, Z.; Flynn, M.P. A 3.43 TOPS/W 48.9 pJ/pixel 50.1 nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS. In Proceedings of the 2017 Symposium on VLSI Circuits, Kyoto, Japan, 5–8 June 2017; pp. C30–C31.
61. Amravati, A.; Nasir, S.B.; Thangadurai, S.; Yoon, I.; Raychowdhury, A. A 55nm time-domain mixed-signal neuromorphic accelerator with stochastic synapses and embedded reinforcement learning for autonomous micro-robots. In Proceedings of the 2018 IEEE International Solid-State Circuits Conference-(ISSCC), San Francisco, CA, USA, 11–15 February 2018; pp. 124–126.
62. Chen, G.K.; Kumar, R.; Sumbul, H.E.; Knag, P.C.; Krishnamurthy, R.K. A 4096-neuron 1M-synapse 3.8-pJ/SOP spiking neural network with on-chip STDP learning and sparse weights in 10-nm FinFET CMOS. *IEEE J. -Solid-State Circuits* **2018**, *54*, 992–1002. [[CrossRef](#)]
63. Kim, D.; She, X.; Rahman, N.M.; Chekuri, V.C.K.; Mukhopadhyay, S. Processing-in-memory-based on-chip learning with spike-time-dependent plasticity in 65-nm cmos. *IEEE Solid-State Circuits Lett.* **2020**, *3*, 278–281. [[CrossRef](#)]