Fall 11-18-2020

MACHINE LEARNING AUGMENTATION MICRO-SENSORS FOR SMART DEVICE APPLICATIONS

Mohammad H. Hasan
University of Nebraska-Lincoln, mohammadhhasan@huskers.unl.edu

MACHINE LEARNING AUGMENTATION MICRO-SENSORS FOR SMART DEVICE APPLICATIONS

by

Mohammad H Hasan

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Doctor of Philosophy

Major: Mechanical and Materials Engineering

Minor: Mathematics

Under the Supervision of Professor Fadi Alsaleem

Lincoln, Nebraska

November, 2020

MACHINE LEARNING AUGMENTATION MICRO-SENSORS FOR SMART DEVICE APPLICATIONS

Mohammad H Hasan, Ph.D.

University of Nebraska, 2020

Adviser: Dr. Fadi Alsaleem

Novel smart technologies such as wearable devices and unconventional robotics have been enabled by advancements in semiconductor technologies, which have miniaturized the sizes of transistors and sensors. These technologies promise great improvements to public health. However, current computational paradigms are ill-suited for use in novel smart technologies as they fail to meet their strict power and size requirements. In this dissertation, we present two bio-inspired colocalized sensing-and-computing schemes performed at the sensor level: continuous-time recurrent neural networks (CTRNNs) and reservoir computers (RCs). These schemes arise from the nonlinear dynamics of micro-electro-mechanical systems (MEMS), which facilitates computing, and the inherent ability of MEMS devices for sensing. Furthermore, this dissertation addresses the high-voltage requirements in electrostatically actuated MEMS devices using a passive amplification scheme.

The CTRNN architecture is emulated using a network of bistable MEMS devices. This bistable behavior is shown in the pull-in, the snapthrough, and the feedback regimes, when excited around the electrical resonance frequency. In these regimes, MEMS devices exhibit key behaviors found in biological neuronal populations. When coupled, networks of MEMS are shown to be successful at classification and control tasks. Moreover, MEMS accelerometers are shown to be successful at acceleration waveform classification without the need for external processors.

MEMS devices are additionally shown to perform computing by utilizing the RC architecture. Here, a delay-based RC scheme is studied, which uses one MEMS device to simulate the behavior of a large neural network through input modulation. We introduce a modulation scheme that enables colocalized sensing-and-computing by modulating the bias signal. The MEMS RC is tested to successfully perform pure computation and colocalized sensing-and-computing for both classification and regression tasks, even in noisy environments.

Finally, we address the high-voltage requirements of electrostatically actuated MEMS devices by proposing a passive amplification scheme utilizing the mechanical and electrical resonances of MEMS devices simultaneously. Using this scheme, an order-of-magnitude of amplification is reported. Moreover, when only electrical resonance is used, we show that the MEMS device exhibits a computationally useful bistable response.

Acknowledgement

I would like to sincerely express my thanks and appreciation to my Ph.D. advisor and the chair of my Ph.D. committee, Dr. Fadi Alsaleem, for his continuous support, insightful feedback and his abundant patience with me since I have joined his research group as a M.Sc. student. This research would not have been possible without his guidance.

I would also like to express my great appreciation to my committee members, Dr. Ndao, Dr. Markvicka, Dr. Alahmad and for my minor advisor, Dr. Velcsov for their insightful comments and remarks and for taking time off their busy schedules to be a part of my Ph.D. committee. Their diverse academic backgrounds have helped me expand my scientific horizons and enrich my research. I would also like to thank the faculty whom I have collaborated with throughout my Ph.D. program, Dr. Younis, Dr. Ouakad, Dr. Ramini, Dr. AbdelRahman, Dr. Jafari and Dr. Pourkamali, as well as their students in their perspective universities. This research would not have been possible without their expertise.

I would like to also express my appreciation to all my friends and colleagues in PKI who have made my graduate school experience a delightful and unforgettable one. I would like to namely thank my best friend MohdEslam for giving me so many useful research tips at the start of my M.Sc. program and for all the fun time we have had together. I would like to also thank my very special friend Dr. Dongfeng who has always supported me and helped me improve my work with her hyper-critical comments. Finally, I would like to thank Mehari and Ali for being the best group members I could ever hope for and for being amazing friends.

Last but certainly not least, I would like to sincerely thank my family for their endless love and support. I would like to express my thank for my father who has always believed in me and pushed me to improve myself, my mother for her boundless love and patience, my older brother and sister, Tareq and Dina, for their support and wisdom and my younger brother and sister, Yahya and Deemah, for being always cheering me on and for being my best friends in the world.

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION AND MOTIVATION

1.1 Dissertation Motivation

Advances in complementary metal-oxide semiconductor (CMOS) technologies have 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 range in size from large (smart cars [1] [2], smart homes [[3] [4] [5]) to minuscule (smartwatches [6] [7] and other wearable devices [8] [9] [10]). Consequently, data is 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 micro-electro-mechanical-systems (MEMS) sensors and fabrication techniques. Processing data in smart devices, however, has proven to be a struggle. On the one hand, larger smart systems require complex analysis of data generated by many sensors such as building management in smart homes. On the other hand, smaller smart systems possess limited size and power capacities, which limit them to either perform simple computation locally or transmit their data to be processed in the cloud.

In all the aforementioned smart devices, i.e. wearable electronics and micro-robots, data processing is considered inefficient size- and power-wise when performed locally. Improvements in local computing power are plateauing as we approach the end of Moore's law[1] due to reaching the physical limits of transistor sizes miniaturization [11] and the complicated thermal management

[1] Moore's law is a projection based on observation of historical trends in transistor production. Contrary to what the name suggests, Moore's law is not a physical law.

requirements of existing CMOS-based logic gates [12] [13]. Latency is another concern as the memory-processor separation throttles the speed of the system. This latency, named the 'von-Neumann bottleneck', is inherent to the computing scheme used in digital computing. In contrast, cloud computing bypasses the size limitation of processors by transferring data for external processing. However, data transmission was shown to constitute a significant amount of power loss [14] [15] and may raise privacy concerns [16]. Furthermore, latency due to data transmission is a big drawback in cloud-computing-based architectures, which makes them ill-suited for real-time processing.

Therefore, there is a great need for new computing architecture suited for 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, 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 [14]. In this dissertation, an analog-based colocalized sensing-and-computing architecture is presented using micro-electro-mechanical-system (MEMS) sensors. Computing is performed immediately at the sensor node to compress analog data into meaningful information (e.g. in a wearable device application concerned with the distance traveled, 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.

To bypass the pitfalls of the von Neumann architecture, MEMS devices in this dissertation employ a non-conventional computing approach named 'neuromorphic computing', which is a neuro-

inspired approach to computing [17]. Various works [18] [19] have shown that neuromorphic approaches to computing can circumvent the limitation of current smart systems that stem from relying on the von-Neumann architecture for digital computing. The predominant neuromorphic computing idea is the use of neuronal structures, due to their structure enabling a high degree of parallel computing and colocalized memory-and-processing [17], and the extreme energy efficiency of their biological counterparts [20]. Information pre-processing at/near the sensor level has also been shown to be effective at addressing these challenges [14]. In what follows, a literature review of recent breakthroughs in non-conventional computing is presented.

1.2 Literature Review

Smart and wearable devices often gather data to perform tasks such as health diagnostics [21], warning generation for adverse events such as elderly falls [22] and/or assisting the ill [23] by relying on specific signatures in the data. These signatures are often complex to extract using classical methods and/or are individual-specific. Consequently, machine learning, often Artificial Neural Networks-based (ANNs), is often employed to perform these tasks. As such, computation via machine learning will be the main focus of this dissertation.

While theoretically suitable for use, machine learning is prohibitively expensive to employ in smaller systems, such as wearable devices and micro-robots. Moreover, due to size and weight restrictions, large ANNs cannot be utilized locally using digital processors. Off-sourcing data-processing using cloud computing results in energy loss via wireless communication [14] [15], possess privacy risks [11] and begets internet-dependence in operation.

The problems with digitally simulating artificial neural networks are expected to persist due to the inevitable end of Moore's law. These limitations mainly arise from the inefficiency of the von-Neumann architecture used in digital processors. In this architecture, memory and processors are separated, which introduces energy losses in the communication buses and latency (von-Neumann bottleneck). Therefore, alternative computational architectures have been introduced, mainly

inspired by computation in living beings. One of the most prominent of these approaches is Neuromorphic Computing.

1.2.1 Neuromorphic Computing

The concept of Neuromorphic computing, a term conceived by Mead [20] in the late 1980s, was introduced to address the limitations of the von-Neumann architecture. 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 expanded to include the use of digital elements and hybrid analog-digital circuits to perform the same goal [17]. Biological neuronal networks are well known for their incredible power efficiency, and ability to solve complex computational problems. This is due to parallelism, colocalized memory, and processing. Indeed, it is said that the human brain is up to nine orders of magnitude more efficient than transistor-based digital computers [20]. Therefore, by utilizing alternative computing architecture based on biological neurons, it may be possible to enhance the computational ability of current processors and enable complex computation in novel smart systems.

Various neuronal models have been developed based on biological neurons [24] from the simple leaky integrate-and-fire neurons to the complex Hodgkin-Huxley [25] and Izhikevich [26] models. It was shown in the literature that even the simplest of neuronal models can be extremely potent when coupled with large networks in a computing architecture named neural networks.

1.2.2 Artificial Neural Networks and Continuous-Time Recurrent Neural Networks

Artificial neural networks (ANN) are visualized as networks of nonlinear nodes (named neurons) arranged into various structures called “layers”. These layers are arranged in a sequential fashion starting from an input layer (first layer) and ending with an output layer (last layer). If the network has more than 2 layers, intermediate layers are named “hidden layers”. Often in the ANN literature,

the number of layers in a neural network is understood as the number of hidden layers as the input and output layers are inherent to ANN structures. It is noted here that simulated neural networks are not typically considered a subset of neuromorphic as they suffer from the von Neumann drawbacks in this form.

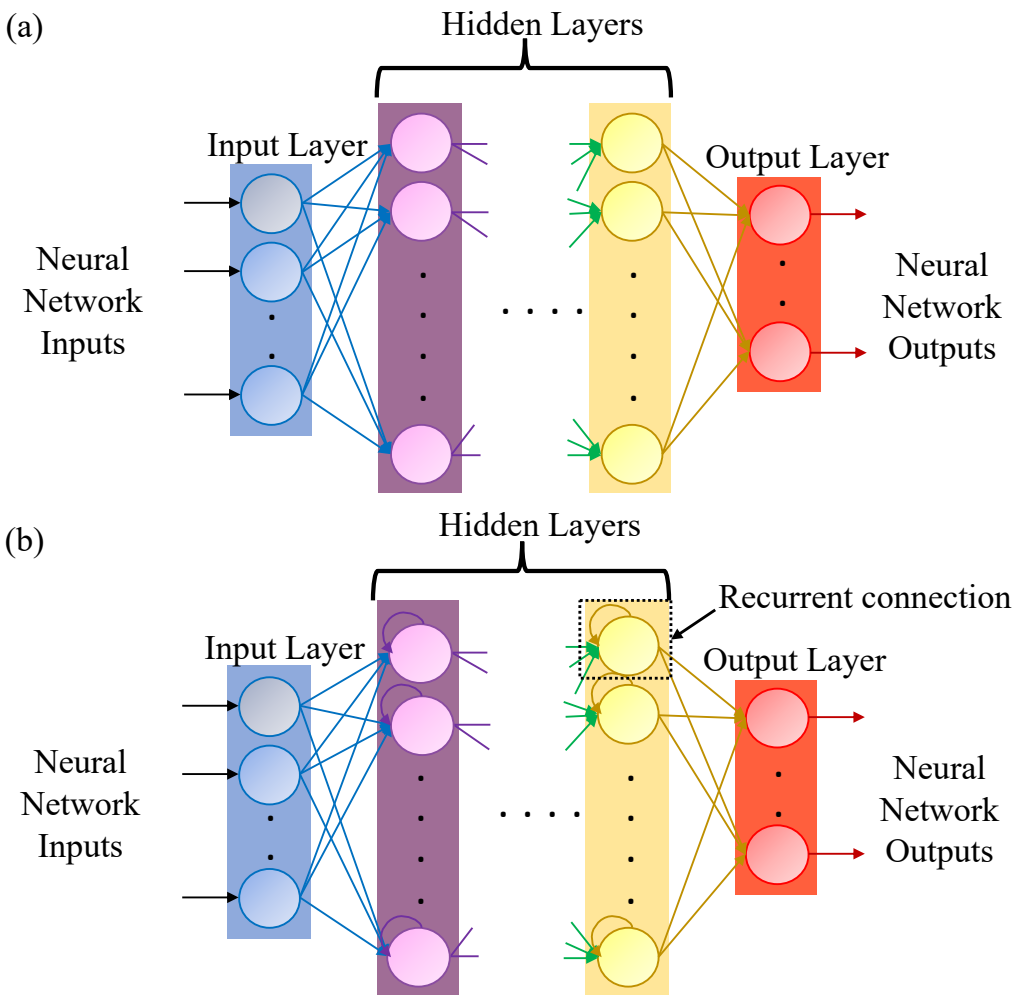


Figure 1.1: Artificial Neural Network schematics showing the input layer, hidden layers and output layers. (a) Feedforward neural networks (FFNN). (b) Recurrent neural network (RNN). Note the self-connection absence in FFNN and addition in RNNs.

In traditional ANNs, named "Feedforward neural networks" FFNN, each neuron can only receive inputs from other neurons in its subsequent layer and can only supply its outputs to neurons of its immediate consequent layer, as shown in FIG.1.1 In this architecture, self-connection (self-

feedback) and inter-layer connections are prohibited. This architecture experiences a critical problem of memory retention as the FFNN reacts to inputs discretely disregarding their past inputs.

To address this limitation, newer architectures of neural networks, named Recurrent neural networks (RNNs) were developed. These networks retain memory through recurrent self-connections, or through connections with neurons within the same layer. Various types of RNNs exist from simple discrete Elman Networks [27] to the popular and computational expensive Long Short-Term Memory (LSTM) neural networks [28]. An RNN architecture of particular interest in this dissertation is the Continuous-Time Recurrent Neural Networks (CTRNNs) [29] which offer higher computational ability than RNNs while requiring less computational resources to model compared to LSTMs. In addition to their balanced stats in computational power and modeling complexity, the continuous nature of CTRNNs is what makes them particularly attractive as it enables them to be emulated by physical devices, such as photonics [30] and sub-threshold transistors [31]. CTRNNs offer additional potential due to their similarity to neurons used in the Dynamic Field Theory [32] [33] , a mathematical theory for understanding human memory.

1.2.3. Reservoir Computing

Unlike FFNNs and RNNs, the reservoir computing (RC) architecture foregoes organizing neurons into layers, as shown in FIG.1.2. Instead, in RC architectures, large numbers of neurons are randomly, yet sparsely, interconnected, thus performing high dimensional transformation using sheer numbers (additional details available in chapter 3). RC schemes typically lack dedicated output neurons. Rather, the RC output is computed via a weighted linear summation of all neuron outputs. Each RC network output requires its own dedicated weighted linear summation unit named reservoir readout.

Similar to RNNs, RCs may retain the memory of previous inputs via recurrent connections, thus enabling them to perform time-dependent tasks, such as time-series analysis. RCs have proven

successful at some computationally difficult benchmark tests such as chaotic series classification [34] [35] and spoken digits classification [36].

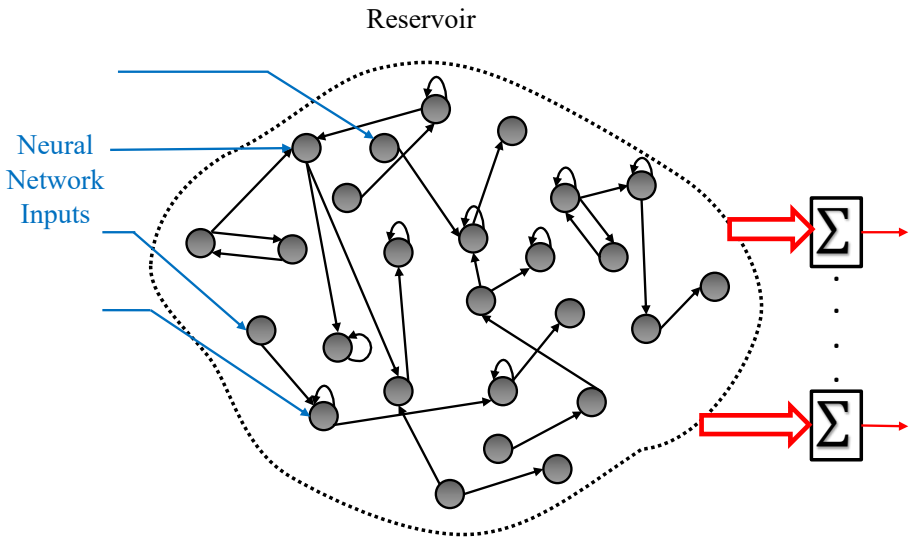


Figure 1.2 Reservoir computing schematics detailing the connections of input signals with the reservoir, the sparse and random connections between neurons within a reservoir (including self-connection), and the connections with the reservoir readout (denoted by a sum block). For the sake of convenience, it is noted here that the thick red arrows connecting the reservoir to the readout circuit signify the outputs of all reservoir neurons.

The origin of RCs can be traced back to the beginning of the 21st century by the names of Echo state networks (ESNs) and Liquid state machines (LSMs), introduced at around the same time independently by Jeager [37] and Maass [38], respectively. RC remains an active research topic with novel research describing the use of single nonlinear analog devices to simulate a large reservoir [39] [40] [41] [42] [43] [44], the use of coupled reservoirs [45] and spatiotemporal multiplexing of RC [46], among other research directions.

Recent research has pointed to the biological plausibility of RC structures, as neuronal structures named ‘mushroom bodies’ [47] that exist in some insects are believed to function in a fashion similar to computational RC structures [48], suggesting the biological plausibility of this architecture. As such, some research has been performed on using RC architectures to control robotic insects [48] and perform classification tasks in soft robots [49].

1.2.4. Applied neuromorphic computing

Neuromorphic computing has been implemented using a variety of methods. Mead [20] originally demonstrated the implementation of artificial neural networks using electronic transistors operated at the sub-threshold regime. Various other works utilized this concept to demonstrate non-digital neural networks [50] [51]. More recently, novel electronic components named memristors (variable resistance elements with a memory through hysteresis) have been extensively employed instead of transistors for neuromorphic computing. Memristive devices have shown great potential as tools to emulate biologically plausible spiking neurons, rather than artificial neural networks, through using memristors to model spike timing dependent plasticity [52] [53].

Artificial neural networks have also been demonstrated using neuromorphic approaches. Physical CTRNNs have been demonstrated using photonics [30] and subthreshold transistors [31], while LSTMs have been emulated using resistive cross-point devices [54] and memristive devices [55]. The physical emulation of artificial and biologically plausible neural networks has become increasingly easier with the development of TrueNorth [56], a programmable neuromorphic chip offering convenience, and high flexibility.

Biologically inspired neural networks have resulted in the development of neuromorphic sensors. In theory, these sensors operate similarly to biological systems, where constant inputs saturate and get ignored while transient inputs are transmitted to the brain. This idea gave rise to the Address Event Representation AER scheme, which is used in neuromorphic vision [57] [58] [59], auditory [60] [61], olfactory [62] and tactile [63] sensors. It is noted here that, while neuromorphic sensors operate quite efficiently, the sensors themselves are not neuromorphic. ‘Neuromorphic sensors’ is a term usually used to describe a traditional sensor, integrated with an AER-based neuromorphic chips, where the location of output-varying sensors (in a sensor array) is transmitted as spikes to a neuromorphic processing chip.

Reservoir computing has also risen into prominence due to the discovery of delay-based RC, which is a scheme that bypasses the strict RC requirement of large numbers of physical neurons by creating virtual neurons in nonlinear dynamical nodes. This concept was first introduced in [39] and was later used a photonics device for analog computation. Delay-based RC was shown to be successful in various other systems such as spin-torque oscillators [64], Boolean nodes [41] and microelectromechanical systems (MEMS) [44].

The current neuromorphic models introduce parallelism and memory-processor colocalization while avoiding the von Neumann shortfalls. However, the current neuromorphic computing approaches for data analysis necessitate the addition of neuromorphic sensors, which increases the size of the smart system overall. This may not be convenient for technologies with limited space requirements. Moreover, new digital-based neuromorphic sensors still require additional circuitry for signal conditioning and analog-to-digital conversion, which represent avenues for energy loss, especially when signals are generated and processed at high speed. Biologically plausible neural networks and neuromorphic sensors require means to interpret spiking signals, further increasing the size of the system. This dissertation departs from the state-of-the-art neuromorphic computing architectures by performing neuromorphic computing at the sensor level, truly reducing the size of smart systems while saving energy by performing neuromorphic computing with very few additional components.

1.3 Dissertation Objectives

The goal of this dissertation is to develop neuromorphic micro-sensors capable of addressing the strict requirement of future technologies (miniature size, low power consumption). To this end, this dissertation aims to depart from the traditional sensing and computing approach in which a physical signal is transformed into an analog electrical signal via a transducer, then translated to a digital signal via analog-to-digital converters (ADC), and finally sent as a string of binary digits (bits) to a digital processor for digital processing.

The neuromorphic micro-sensors developed in this dissertation also differ from other prevalent neuromorphic sensors in the fact that information is processed in analog at the sensor level, rather than transforming data into a spiking signal that is later processed using a spiking neural network (SNN) embedded in a neuromorphic processor. The elimination of the additional components in SNN is essential in some applications that require extreme miniaturization, such as micro-robotics and wearable devices, and in applications that require minimal rigid electronics, such as flexible electronics and soft robotics.

To this end, the neuromorphic sensing-and-computing in this work is colocalized in the sensor node. This dissertation mainly focuses on nano/microelectromechanical systems (N/MEMS) for colocalized sensing and computing due to their prevalence in smart systems, their convenience in use, and their nonlinear dynamics. Specifically, this dissertation objective is addressing the following goals:

G1: Finding the dynamical behaviors that facilitate computing in general and how to generate them in MEMS sensors.

G2: Developing schemes to couple micro-sensors to enable computing through analog neural networks (CTRNNs and RCs). Furthermore, this dissertation modifies the developed schemes to allow for colocalized sensing-and-computing.

G3: Demonstrating the particular advantages of MEMS devices for colocalized sensing-and-computing.

An additional research objective in this dissertation involves studying the practicality of using electrostatic MEMS devices for colocalized sensing-and-computing. Mainly, this dissertation addresses modeling the response changes of electrostatic MEMS devices at very high frequencies. Furthermore, we address the high voltage requirements to actuate general electrostatic MEMS devices by proposing a novel actuation scheme named ‘Double resonance drive’. (G4)

The organization of this dissertation is as follows: In Chapter 2, the dynamics of MEMS are introduced, in chapter 3, a background in neural networks and neuromorphic computing is presented, in chapter 4, the dynamics of MEMS devices utilized to perform some simple computational tasks in an attempt to address G1, the MEMS devices are then coupled in a large network in chapter 5 to perform a benchmark test. Chapter 6 addresses reservoir computing using MEMS sensors and introduces a modified RC approach to facilitate colocalized sensing and computing to address G2. Chapter 7 sheds some light into additional nonlinear dynamics in MEMS devices that can be utilized to further improve MEMS performance both in sensing and computation, thus posing some answers to Q3. Chapter 8 discusses some potential future directions for this research. Finally, chapter 9 presents an overall conclusion to this dissertation. This thesis is supplemented with experimental results throughout chapters 4, 6 and 7.

It is noted here that, in principle, the colocalized neuromorphic sensing and computing schemes discussed in this work are suitable for various types of sensors aside from electrostatic MEMS devices. The operating regime and/or modifications necessary to facilitate colocalized sensing and computing may differ as well. However, provided the sensors themselves satisfy the conditions necessary for neuromorphic computing (such as the RC requirements), colocalized sensing-and-computing may be possible.

Chapter 2
INTRODUCTION TO MEMS DYNAMICS

Microelectromechanical systems (MEMS) appear in the literature in various forms. Structurally, they can be as simple as microbeams (FIG.2.1,a) or as complex as compounded structures and Combdrive devices (FIG.2.1,b). MEMS devices also differ in their means of excitation, with electrostatic excitation and piezoelectric excitation being the two most popular means of excitation. In this dissertation, mainly beam-structure, electrostatically actuated MEMS devices are considered. This chapter briefly introduces the dynamics of MEMS devices.

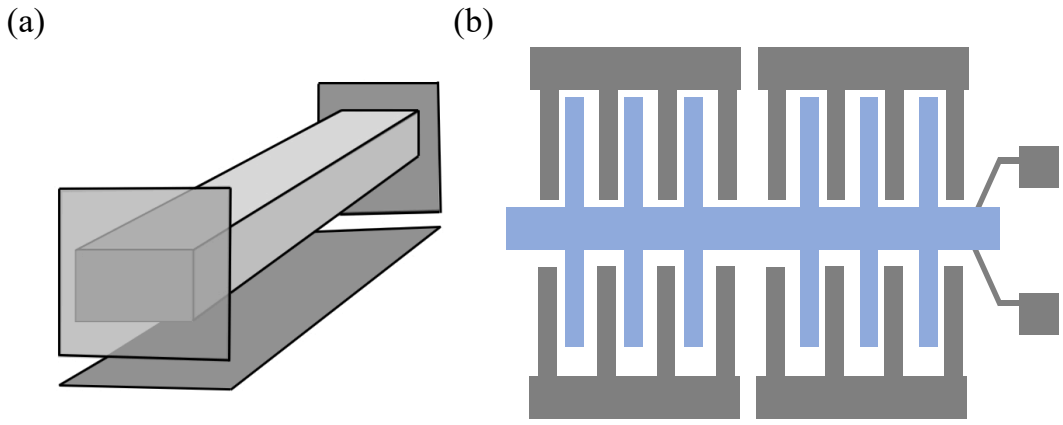


Figure 2.1 MEMS schematics. (a) MEMS microbeam (silver). (b) MEMS Combdrive. The moving proof mass is colored blue.

2.1 Introduction of MEMS dynamics

MEMS devices may be modeled as single-degree-of-freedom (SDOF) systems, such that the entirety of the MEMS inertia is assumed to concentrate into a single point mass, the structural elasticity is modeled as spring element (linear or nonlinear), and the energy dissipation is modeled as a damper element. In this case, the simplified model only solves the structural deflection at a single point.

Alternatively, MEMS devices may be modeled as continuous microbeams, where the deflection of the microbeam can be computed for every point along with the MEMS structure. The continuous

microbeam model of MEMS devices is more accurate than the SDOF model. However, SDOF models are computationally efficient to study, especially in applications such as simultaneous sensing-and-computing in this work where multiple MEMS devices are simulated simultaneously, or when MEMS devices are actuated for a relatively long period compared to the integration step size. While both the SDOF model and the continuous model are used in this dissertation, the former is used more often for convenience.

2.2.1 SDOF MEMS dynamics

The dynamics of a SDOF electrostatically actuated MEMS device is governed by a first-order spring-mass-damper system equation (2.1) and a schematic of a SDOF MEMS device is shown in FIG.2.2:

$$m_{eff}\ddot{x}(t) + c_{eff}\dot{x}(t) + k_{eff}x(t) = F_e(x,t) \tag{2.1}$$

Where m_{eff}, c_{eff} and k_{eff} are the effective mass, damping coefficient and stiffness coefficient of the microbeam, respectively, $x(t)$ is the deflection of the proof mass at time t; the dot operator in (2.1) represents temporal derivatives, and $F_e(x,t)$ is the electrostatic force given by (2.2):

$$F_e(x,t) = \frac{\varepsilon b L V_{MEMS}(t)^2}{2(d-x)^2} \tag{2.2}$$

The microbeam has a length and width given by b and L, respectively. The vibrating microbeam (moving electrode) and the stationary electrode are separated by a dielectric material with a permittivity ε and a thickness d. The electrostatic force is generated due to a constant or time-varying potential difference (total applied voltage) $V_{MEMS}(t)$, which may be AC, DC, or a combination of both.

The predominant form of damping is squeeze-film damping. A mechanism through which kinetic energy is lost via compressing the air between the moving and stationary electrodes (viscous damping). The nonlinear squeeze film damping is governed by (2.3):

$$c(x) = \frac{64\sigma(x)P_a A_s}{2\pi^7 f(d-x)} \frac{1+\beta^2}{(1+\beta^2)^2+\frac{\sigma(x)^2}{\pi^4}} \tag{2.3}$$

where $c(x)$ is the nonlinear squeeze film damping, $\sigma(x)$ is the squeeze number, given by (3.4), P_a is the ambient pressure, $A_s = bL$ is the MEMS surface area, f is the vibration frequency, and β is the MEMS aspect ratio.

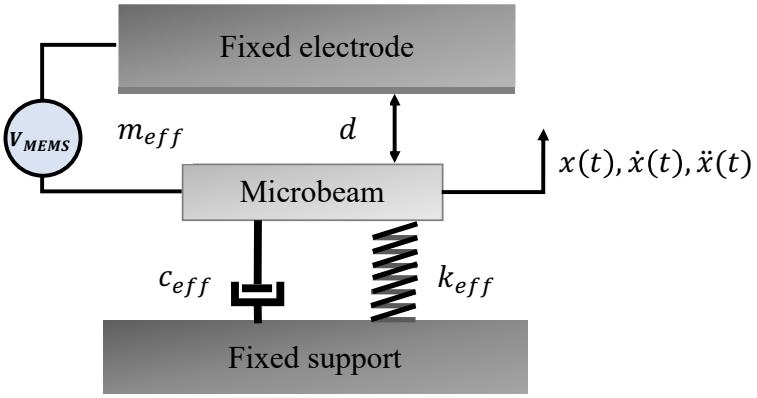


Figure 2.2 SDOF MEMS schematics.

$$\sigma(x) = \frac{12\, A_s\, \mu_{eff}}{P_a\, (d-x)^2} \tag{2.4}$$

where μ_{eff} is the effective viscosity of air, accounting for the slip boundary condition, given by (2.5) [65] [66].

$$\mu_{eff} = \frac{\mu}{1+9.638Kn^{1.159}} \tag{2.5}$$

where μ is the nominal viscosity of air at the given temperature and humidity conditions and Kn is the Knudsen number calculated using (2.6) and (2.7), sequentially.

$$\lambda_a = \frac{\lambda_0 P_0}{P_a} \tag{2.6}$$

$$Kn = \frac{\lambda_a}{d} \tag{2.7}$$

where λ_0 and λ_a are the mean-free path of gas molecules at atmospheric pressure P_0 and at the operating pressure, respectively.

2.2.2 Microbeam dynamics

The dynamics of a microbeam arch are governed by the Euler-Bernoulli equation (2.8):

$$EI_c \frac{\partial^4 w(x,t)}{\partial x^4} + \rho A_{cs} \frac{\partial^2 w(x,t)}{\partial t^2} + c \frac{\partial x(x,t)}{\partial t} =$$

$$\left(\frac{\partial^2 w(x,t)}{\partial x^2} - \frac{d^2 w_0(x)}{dx^2}\right)\left[\frac{EA_{cs}}{2L}\int_0^L \left\{\left(\frac{\partial w(x,t)}{\partial x}\right) - 2\left(\frac{\partial w(x,t)}{\partial x}\frac{dw_0(x,t)}{dx}\right)\right\} dx\right] - f_e(x,t) \qquad (2.8)$$

where $w(x,t)$ is the deflection of the microbeam at a distance x from the support at time t; ρ, E, I_c and A_{cs} are the mass density of the microbeam, the Young modulus of elasticity, the second moment of area and the cross-sectional area, respectively. c is the damping constant per unit length, $w_0(x)$ is the nominal elevation due to curvature at x and $f_e(x,t)$ is the electrostatic force per unit length. For a clamped-clamped MEMS microbeam, like the one shown in FIG.2.3, the following boundary conditions are applied (assuming perfect boundaries):

$$w(0,t) = w(L,t) = 0 \qquad (2.9\text{-a})$$

$$\frac{\partial w(0,t)}{\partial t} = \frac{\partial x(L,t)}{\partial t} = 0 \qquad (2.9\text{-b})$$

For convenience, equation (2.8) is normalized using the following set of nondimensional variables:

$$\hat{x} = \frac{x}{L},\ \widehat{w} = \frac{w}{d},\ \widehat{w_0} = \frac{w_0}{d},\ \hat{t} = \frac{t}{T} \qquad (2.10)$$

Substituting the nondimensional variables into (2.10) yields (2.11):

$$\frac{\partial^4 w(x,t)}{\partial x^4} + \frac{\partial^2 w(x,t)}{\partial t^2} + \tilde{c} \frac{\partial x(x,t)}{\partial t} =$$

$$\alpha_1\left(\frac{\partial^2 w(x,t)}{\partial x^2} - \frac{d^2 w_0(x)}{dx^2}\right)\left[\int_0^L \left\{\left(\frac{\partial w(x,t)}{\partial x}\right) - 2\left(\frac{\partial w(x,t)}{\partial x}\frac{dw_0(x,t)}{dx}\right)\right\} dx\right] - \alpha_2 f_e(x,t) \qquad (2.11)$$

With the following boundary conditions for a clamped-clamped MEMS:

$$w(0,t) = w(1,t) = 0 \qquad (2.12\text{-a})$$

$$\frac{\partial w(0,t)}{\partial t} = \frac{\partial x(1,t)}{\partial t} = 0 \tag{2.12-b}$$

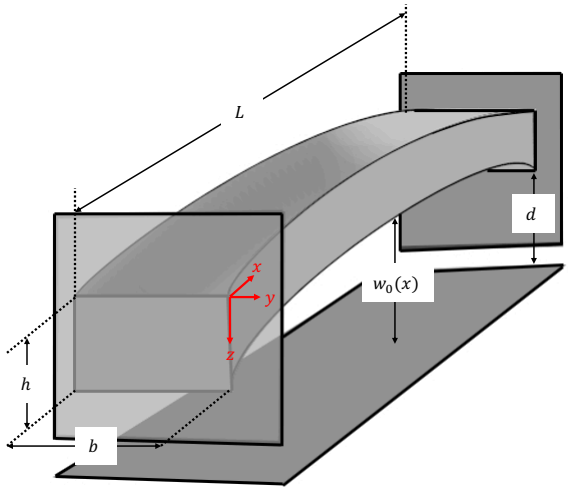


Figure 2.3 Schematics of a MEMS clamped-clamped arch beam.

Noting that the hat is removed for convenience. The new normalized constants are given by (2.13-2.16) and further noting that the normalized electrostatic force per unit length is given by (2.17) [67]

$$T = \sqrt{\frac{\rho A_{cs} L^4}{E I_c}} \tag{2.13}$$

$$\alpha_1 = 6\left(\frac{d}{h}\right)^2 \tag{2.14}$$

$$\alpha_2 = \frac{\varepsilon b L^4}{2 E I_c d^3} \tag{2.15}$$

$$\tilde{c} = \frac{c L^4}{E I_c T} \tag{2.16}$$

$$f_e = \frac{V_{tot}^2}{(1+w_0+w)^2} \tag{2.17}$$

where α_1 is the stretching parameter, α_2 is the forcing parameter, $\tilde{c}$ is the damping parameter, and T is the normalization time.

Solving (2.11) requires intensive computation. A popular approach way to simplify this process is using a simple separation of variables via the Galerkin method:

$$w(x,t) = \sum_i^{\infty} \varphi_i(x) u_i(t) \tag{2.18}$$

Here, the deflection of the microbeam is given by a special component $\varphi_i(x)$ known as the modeshape and a temporal component named the modal coordinate $u_i(t)$. Theoretically, the response of a microbeam is the outcome of an infinite sum of $\varphi_i(x)u_i(t)$. However, in practice, a finite number of modeshapes is sufficient to reach any required degree of accuracy. It is noted here that the modeshapes are computed based simply by solving the Eigenvalue problem of the MEMS device, while computing the modal coordinate requires more involved work.

The Eigenvalue problem arises from solving the steady-state, unforced response of the MEMS device:

$$\frac{\partial^4 w(x,t)}{\partial x^4} = \alpha_1 \left(\frac{\partial^2 w(x,t)}{\partial x^2} - \frac{d^2 w_0(x)}{dx^2}\right)\left[\int_0^L \left\{\left(\frac{\partial w(x,t)}{\partial x}\right) - 2\left(\frac{\partial w(x,t)}{\partial x}\frac{dw_0(x,t)}{dx}\right)\right\} dx\right] \tag{2.19}$$

By taking (2.18) into account and by setting $u_i(t) = a_i$ (steady state), solving (2.19) for $\varphi(i)$ yields an infinite number of non-trivial solutions. These infinite solutions correspond to the microbeam modeshapes φ_i and their corresponding non-dimensional modal frequencies ω_i.

For a simple straight clamped-clamped MEMS device, the microbeam modeshapes are given by (2.20) [68]:

$$\phi_i(x) = \cosh\left(\sqrt{\omega_i}x\right) - \cos\left(\sqrt{\omega_i}x\right) - \psi_i\left[\sinh\left(\sqrt{\omega_i}x\right) - \sin\left(\sqrt{\omega_i}x\right)\right] \tag{2.20}$$

Where $(\omega_1, \psi_2) = (22.37, 0.98)$, $(\omega_2, \psi_2) = (61.67, 1.0)$, …, are computed numerically.

The dynamical solution of the microbeam response requires solving (2.11) by relying on (2.18) and the modeshapes computed in the Eigenvalue problem. Therefore, (2.21) is to be solved:

$$\sum_i \varphi_i^{(iv)}(x) u_i(t) + \sum_i \varphi_i(x) \ddot{u}_i(t) + \tilde{c} \sum_i \varphi_i(x) \dot{u}_i(t) =$$

$$\alpha_1 \left(\sum_i \varphi_i''(x)u_i(t) - \frac{d^2w_0(x)}{dx^2}\right)\left[\int_0^L \left\{\sum_i \varphi_i'(x)u_i(t) - 2\left(\sum_i \varphi_i''(x)u_i(t)\frac{dw_0(x,t)}{dx}\right)\right\}dx\right]$$

$$-\frac{\alpha_2 V_{MEMS}^2}{(1+w_0+\sum_i \varphi_i(x)u_i(t))^2} \tag{2.21}$$

Equation (2.21) may lead to unstable numerical solution due to the possibility of a singular solution arising from the final term on the right-hand side of the equation. To avoid this problem, (2.21) is multiplied by $(1 + w_0 + \sum_i \varphi_i(x)u_i(t))^2$. To further simplify solving this equation, we further multiply (2.21) by $\varphi_i(x)$ and integrate over the range [0,1], making use of the modeshape orthonormality:

$$\int_0^1 \varphi_i(x)\varphi_j(x)dx = \delta_{ij} \tag{2.22}$$

where δ_{ij} is the Kronecker delta function, returning 1 if $i = j$ and 0 otherwise. This results in a system of coupled ordinary differential equations in the form:

$$M_{ij}\ddot{u}_i(t) + C_{ij}\dot{u}_i(t) + K_{ij}u_i(t) = F_j(t),\ i = ,1,2,\dots,M,\ j = 1,2,\dots,M \tag{2.23}$$

where M_{ij}, C_{ij} and K_{ij} are the mass matrix, damping matrix and the stiffness matrix, respectively, and F_j is the forcing vector applied to each modeshape. The M ordinary differential equations are solved simultaneously to find $u_i(t)$ at each timestep. Finally, the microbeam response $w(x,t)$ is computed at any point x and any time t using (2.18).

2.2. Nonlinear Dynamics of MEMS

The nonlinear inverse-squared electrostatic forcing, the microbeam curvature and the nonlinear midplane stretching contribute to the complex nonlinear response of MEMS. These nonlinear dynamics were traditionally viewed to be detrimental for the response of MEMS sensors. However, recently, nonlinear dynamics have been found to be beneficial to enhance sensor performance [69] [70] [71] or enable a new behavior, such as switching [72] [73]. In the context of this dissertation,

nonlinear dynamics will be used to enable high dimensional (nonlinear) mapping to facilitate computing.

2.2.1. Pull-in instability

The most crucial nonlinearity arising in microsystems is the pull-in instability, which occurs when the electrostatic force exceeds the spring restoration force, causing the microbeam to contact the stationary electrode and stick into it, as shown in FIG.2.4. For a SDOF MEMS operated quasi-statically (using a DC voltage), the pull-in voltage V_{PI} is computed using (2.24) [68]:

$$V_{PI} = \sqrt{\frac{8k_{eff}d^3}{27\varepsilon A_s}} \tag{2.24}$$

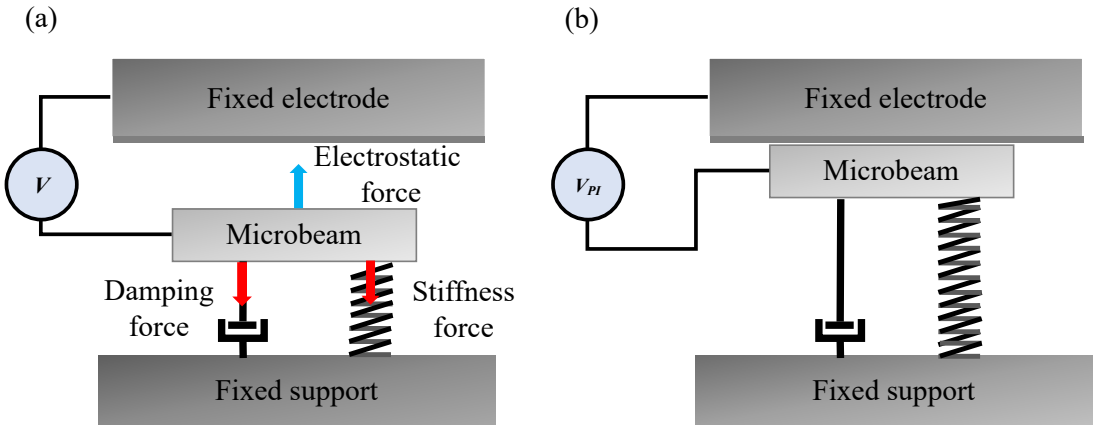


Figure 2.4. Pull-in schematics. (a) The forces acting on the microbeam. The attractive electrostatic force is countered by the stiffness force and the damping force (in dynamic response). (b) The phenomenon of pull-in occurring when the electrostatic force exceeds the restoration force.

Interestingly, if the MEMS device is pulled-in, it cannot be pulled-out (released) unless the input voltage is reduced to $V = V_R < V_{PI}$. As perfect contact between the microbeam is not possible in microscopic sense, the electrostatic force will not be truly infinite upon pull-in as the electrostatic model presented in this work may suggest. To incorporate this fact into this dissertation, one may consider the presence of a rigid dielectric material, with a thickness s, at the surface of the fixed electrode (or the microbeam) that prevents actual electrical contact between the electrodes (this

dielectric layer can be seen in FIG.2.4 as a dark, thin strip at the fixed electrode). In this case, the static release voltage V_R is given by [68]:

$$V_R = \sqrt{\frac{2k_{eff}s^2}{\varepsilon A_s}(d-s)} \tag{2.25}$$

The unequal V_{PI} and V_R will prove important in the forthcoming chapters under the name 'Hysteresis'.

2.2.2. Snapthrough of MEMS arches

MEMS arches also exhibit hysteresis outside the pull-in regime by virtue of their initial curvature. For an appropriate initial elevation $w_0(x)$, the microbeam may buckle about the undeformed axis, experiencing snapthrough. This behavior is demonstrated in FIG.2.5. Prior to snapthrough, the MEMS stiffness is reduced suddenly leading to a sudden response jump. Afterwards, the microbeam stiffness suddenly increases, reducing further deflections and stabilizing the microbeam. One may model this behavior as a dual-spring-mass system, as shown in FIG.2.6 [74]. It should be noted here that large initial elevations will result in an immediate pull-in without experiencing a stable snapthrough response. Snapthrough will be discussed more thoroughly in chapter 4.

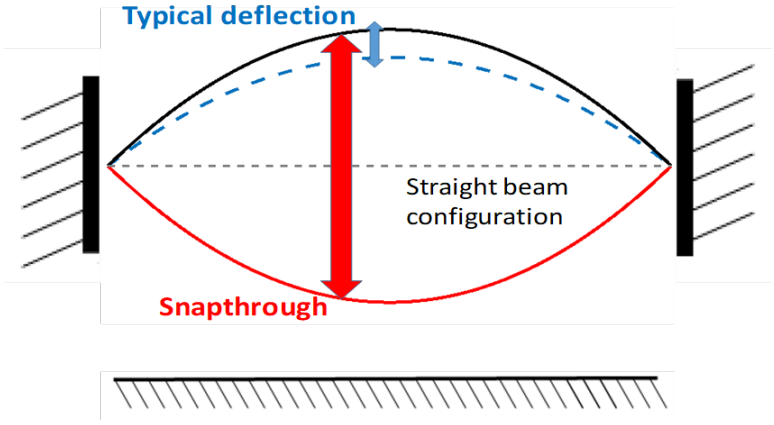


Figure 2.5 Snapthrough schematics experienced by a MEMS arch.

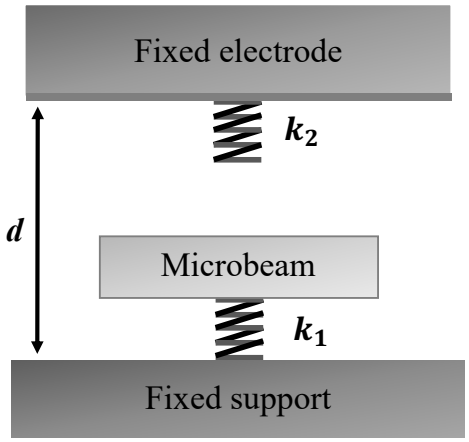


Figure 2.6 A Hysteresis model for visualization. As the input voltage increases, k_1 decreases (softening) resulting in a large response jump. The response jump is stopped once the microbeam reaches the snapthrough configuration. At that moment, the MEMS stiffness increases, symbolized by the microbeam contacting the spring k_2.

2.2.3. Modeling stoppers

The electrostatic forces acting on a MEMS device are inverse-square forces, thus, as the electrostatic gap goes to zero, the electrostatic force approaches an infinite value. This results in unstable integration and exploding values. To address this problem, stoppers are installed in the MEMS system, as shown in FIG.2.7. In such a MEMS system, the microbeam experiences four types of forces: (1) The electrostatic force between the moving and stationary electrode, (2) the stiffness restoration force of the spring, (3) the damping force of the damper, and (4) some reaction force from the stopper, assuming physical contact, as shown in FIG.2.8.

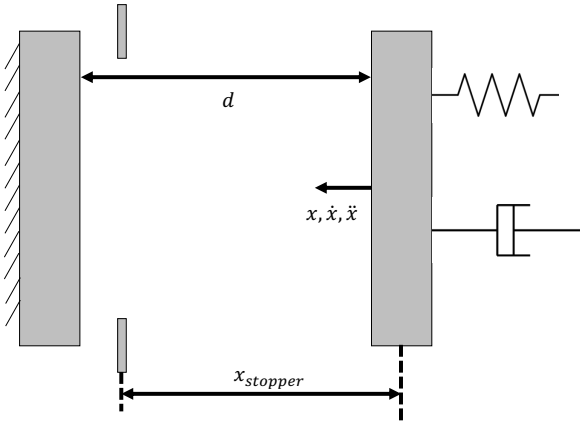


Figure 2.7 Schematics of a SDOF MEMS device with a stopper

The reaction force is only active when the MEMS electrode reaches the stopper located at $x_{stopper}$. Per design, this occurs when the electrostatic force exceeds the sum of the spring and damper forces. At that point, the stoppers will exert a non-positive force on the MEMS device to stop its motion.

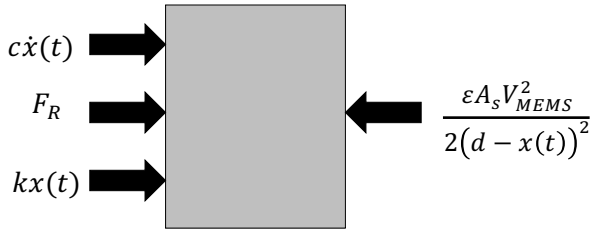


Figure 2.8 Forces acting on the MEMS microbeam

The MEMS equation of motion can be modified to (2.26)

$$m_{eff}\ddot{x}(t) + c_{eff}\dot{x}(t) + k_{eff}x(t) = \frac{\varepsilon A_s V_{MEMS}^2}{2(d-x(t))^2} - F_R \tag{2.26}$$

The reaction force F_R can be found by assuming the microbeam immediately stops when it reaches the stopper. Noting here that this is not very accurate as this assumption does not account for the MEMS momentum and recoil. However, as the goal here is to model the equilibrium point of the MEMS device past pull-in, this approach is considered.

$$\ddot{x} = 0, \dot{x} = 0 \tag{2.27}$$

Therefore, by force equilibrium, the reaction force is found using (2.28)

$$F_R = \frac{\varepsilon A_s V_{MEMS}^2}{2(d-x(t))^2} - k_{eff}x(t) \tag{2.28}$$

But this force only applies when the microbeam touches the support. Thus, this force must include a step function $U(.)$

$$F_R = \left[\frac{\varepsilon A_s V_{MEMS}^2}{2(d-x(t))^2} - k_{eff}x(t)\right] \times U\left(x - x_{stopper}\right) \tag{2.29}$$

Finally, this force must always be positive (multiplied by negative in the EOM to overall be negative). To do this, we also apply another step function:

$$F_R = \left[\frac{\varepsilon A_s V_{MEMS}^2}{2(d-x(t))^2} - k_{eff}x(t)\right] \times U\left(x - x_{stopper}\right) \times U\left(\frac{\varepsilon A_s V_{MEMS}^2}{2(d-x(t))^2} - k_{eff}x(t)\right) \tag{2.30}$$

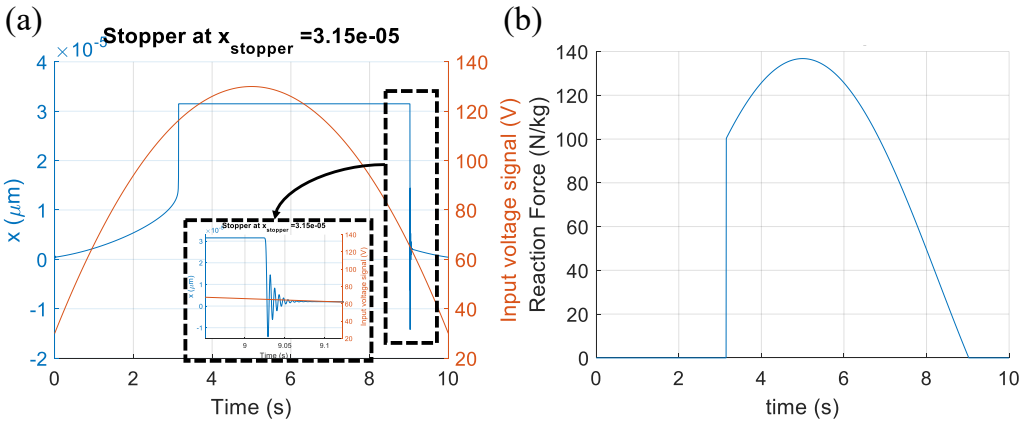


Figure 2.9 Simulated MEMS response utilizing the pull-in simulation scheme described in section 2.2.3. (a) A DC voltage signal is applied to the MEMS device (red). As a response, the MEMS device deflects towards the stationary electrode. At 116V, the MEMS microbeam contacts the stopper. The MEMS microbeam is released from the stopper when the voltage falls below 65V. Insert: zoomed view showing the transience after release. (b) Reaction force corresponding to (a). The reaction force ceases upon release.

The following condition must be imposed on the MEMS equation of motion in the modified model:

$$\dot{x}(t) = \begin{cases} \dot{x}(t), & x(t) < x_{stopper} \\ 0, & x(t) \geq x_{stopper} \end{cases} \tag{2.31}$$

Another possible condition is:

$$\dot{x}(t) = \begin{cases} \dot{x}(t), & F_R < 0 \\ 0, & F_R \geq 0 \end{cases} \tag{2.32}$$

This algorithm for pull-in analysis is tested through simulations, as shown in FIG.2.9. The MEMS dimensions used are given in Table 2.1. The simulation shows a pull-in voltage V_{PI} and a release voltage V_R of 116V and 65V, respectively, which are the same voltages calculated using (2.24) and (2.25).

2.3. Bifurcation and Chaos

The nonlinearities in the MEMS system may result in unstable configurations, which may negatively impact the response of the designed sensing-and-computing device. For instance, if a device is designed to utilize snapthrough for computation (as is the case in chapter 4), it should avoid pulling-in. Therefore, stability analysis is essential. Moreover, studying order and chaos is just as important as specific regimes of stability, such as the edge of chaos [75] are particularly attractive to use in computation while highly ordered regimes and highly chaotic regimes are, in general, ill-suited for computing, owing to the lack of sufficient dynamical complexity and lack of resistance to noise, for each, respectively.

Table 2.1: MEMS parameter for SDOF MEMS in FIG.2.9

Parameter	Value
L	$9\ mm$
b	$5.32\ mm$
d	$42\ \mu m$
k_{eff}	$215\ N/m$
c_{eff}	$0.0711\ N.s/m$
m_{eff}	$146.97\ mg$

A generic dynamical system is given by (2.33,a and 2.33,b):

$$x(t+1) = f(t, x(t)) \tag{2.33-a}$$

$$\dot{x}(t) = f(t, x(t)) \tag{2.33-b}$$

Where $x(t)$ here is the state of the dynamical system at time t and $f(.)$ is an update function. Dynamical systems may be discrete or continuous. In both cases, dynamical systems rely on their previous state to yield future states.

The simplest behavior of a dynamical system is a fixed (equilibrium) point. In this case, the system response is represented by (2.34,a and 2.34,b):

$$x(t+1) = x(t) = X \tag{2.34-a}$$

$$\dot{x}(t) = 0 \tag{2.34-b}$$

Where X is the dynamical system state at the fixed point. The stability of the fixed point is assessed through analyzing the change in $f\big(t, x(t)\big)$ as follows [76]:

If $\left|\frac{df}{dx}\right|(X) < 0$, then the fixed point X is called a stable fixed-point, or an attractor. Otherwise, if $\left|\frac{df}{dx}\right|(X) > 0$ then the fixed point is an unstable fixed-point, or a repeller.

For a second order MEMS system, the fixed points and their stability can be studied using the following state-space approach:

Let $x(t) = x_1(t)$ and $\dot{x}(t) = x_2(t)$. Then, one can write the following system of equations based on (2.1):

$$\dot{x}_1(t) = f_1(x_2(t), t) = x_2(t) \tag{2.35-a}$$

$$\dot{x}_2(t) = f_2(x_1(t), x_2(t), t) = \frac{1}{m_{eff}}\Big(-k_{eff}x_1(t) - c_{eff}x_2(t) + F_e(x_1, t)\Big) \tag{2.35-b}$$

The fixed point must $f_1(X_1, X_2) = f_2(X_1, X_2) = 0$. The stability of the fixed points X_1 and X_2 is checked by finding the Eigenvalues λ of the Jacobian matrix (2.36):

$$\nabla_x F = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{pmatrix} \tag{2.36}$$

If all λ of the fixed point are negative, the fixed point is stable. If at least one λ of the fixed point (X_1, X_2) is positive, the fixed point is unstable.

It is useful for this dissertation to look at the parameter-space of MEMS devices to find the areas of stability and instability, and to distinguish chaos and order. In this case, one may look at a generic discrete dynamical system in the form:

$$X(t+1) = F(X(t), \gamma) \tag{2.37}$$

Where γ is a parameter of interest. For a MEMS device, this parameter may be the mass, stiffness, damping, voltage amplitude, or some other parameters. The system is solved for the fixed points as well as other solutions, such as periodic orbits (oscillations), analytically or numerically. The solutions are plotted against the parameter of interest γ to produce the bifurcation diagram. FIG. 2.10 presents an example of a well-known bifurcation diagram of the logistic map. The dynamical system starts with a single fixed point at $a < 1$ then splits into two solutions: a stable solution (top branch) and unstable solution (bottom branch). As a increases, the dynamical system experiences multiple successive period-doubling bifurcations ending in chaos.

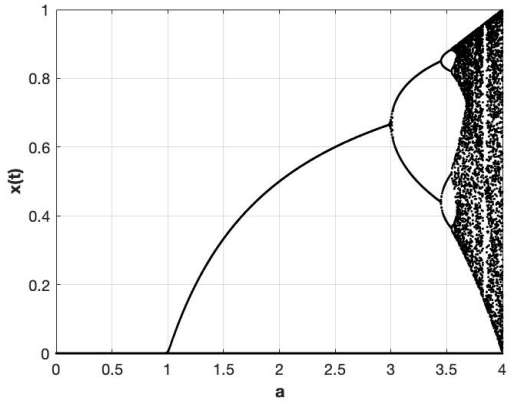


Figure 2.10 Bifurcation diagram of the Logistic map $x(t+1) = ax(t)[1 - x(t)]$ showing the evolution from order to chaos as the parameter a changes.

In addition to the bifurcation diagram, the Lyapunov exponent is required to assess ascertain chaos is present at some value of γ. The Lyapunov exponent (LyE) is a measure of the change in the dynamical system response due to small perturbation. Systems with positive LyE amplify perturbations while systems with negative LyE attenuate them. Thus, a system is chaotic if:

1. The system is aperiodic,
2. LyE is positive.

For a discrete map, such as (3.26-a), LyE of the orbit $\{x_1,\ x_2, \dots\}$ is formulated as

$$LyE(x_1) = \lim_{n\to\infty} \frac{1}{n}\left[\ln\left(f'(x_1)\right) + \ln\left(f'(x_2)\right) + \cdots + \ln\left(f'(x_n)\right)\right] \tag{2.38}$$

Noting that $x_2 = x(2) = f\left(x(1)\right)$.

For the continuous map case, the differential equation (2.33-b) is discretized through numerical integration. Equation (2.38) is then applied.

Chapter 3

NEURAL NETWORKS AND RESERVOIR COMPUTING

3.1. Feedforward Neural Networks

Feedforward neural networks (FFNNs) are the simplest forms of neural networks (Fig.1). The dynamics of simple FFNN of N neurons can be expressed using (3.1)

$$y_i(k+1) = \sigma\left(\sum_{j=1}^{N} w_{ij} y_j(k) + \theta_i + \sum_{j=1}^{P} w_j I_j(k)\right) \quad (3.1)$$

Where $y_i(k)$ is the state of the neuron i at the k timestep, w_{ij} is the coupling weight from the j neuron to the i neuron, θ_i is a constant bias, I_j is the input to the j input neuron, and $\sigma(.)$ is an activation function. Sigmoidal functions are often used as they are smooth, differentiable functions with a finite range (0,1).

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (3.2)$$

Where $\sigma(x)$ is a sigmoidal function.

FFNNs are structured in layers with unidirectional connections, starting from the input layer with P neurons and ending with the output layer with O neurons. Connections are formed strictly from neurons of the M layer to the neurons of the $M+1$ layer, thus $w_{ij} = 0, \forall i \leq j$. This connectivity, along with the governing equation of FFNNs, makes them ill-suited for time-series analysis.

FFNN are optimized by minimizing the cost function of the neural network output through training w_{ij} within the network in a process called ‘backpropagation’. Training neural networks is a highly extensive computational task as the network size grows.

3.2. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) were developed to enable time-series analysis. In this architecture, neurons are organized in layers. However, neurons are allowed to self-connect. One of the simplest RNNs is the Elman network. The Elman network equation follows (3.2) while allowing for self-connection, thus $w_{ij} \neq 0$ for some $i = j$. Despite its simplicity, Elman networks were shown to be effective as units in deep neural networks [77], compared to their computationally expensive counterparts.

Despite its simple structure and effectiveness, this network is both discrete and lacking inertia, unlike physical devices. Instead, this dissertation considers a continuous counterpart to the Elman network named "Continuous-Time Recurrent Neural Networks" (CTRNNs). The dynamics of a CTRNN are given by (3.3)

$$\tau_i \dot{y}_i(t) = -y_i(t) + \sum_{j=1}^{N} w_{ij}\, \sigma\left(y_j - \theta_j\right) + I_i(t) \tag{3.3}$$

Where $y_i(t)$ is the state of the i continuous-time recurrent neuron (CTRN) at time t, τ_i is the time constant representing inertial, and $I_i(t)$ is the input. While the change may seem small, the introduction of τ has been shown to change the orbit of the neural network response [78], enable inherent input signal averaging, and control the memory of the neural network; allowing the CTRNN to react to relatively long input signals [79]. The qualitative similarity between Equation (3.3) and the MEMS dynamics will be explored in Chapter 4.

Training Elman RNNs and CTRNNs is a difficult task involving a process named backpropagation through time. In this dissertation, CTRNNs are trained through genetic algorithms to simplify the process of optimization. This, however, may also increase the optimization duration.

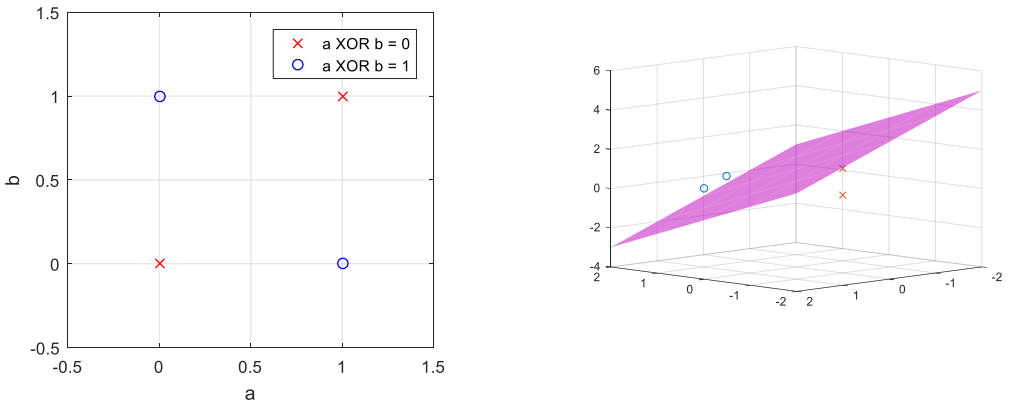
3.3. Reservoir Computing

Reservoir computing (RC) is a computational scheme that utilizes the high-dimensional transformation due to the interactions within a large network of sparsely, yet randomly,

interconnected neurons. Computation through high-dimensional projection can be best visualized through the "XOR problem". In this problem, a classifier is tasked with learning the response of an XOR Boolean logic gate ($\oplus$) (Table 3.1) and Fig.3.1(a). It can be proven that a single linear is insufficient for this classification. However, if the response variable ($a \oplus b$) is mapped into a higher dimensional space, such as projection into 3D space, s single linear classifier (a plane) can be used for classification.

Table 3.1: Truth table of XOR problem

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0



(a) (b)

Figure 3.1 XOR problem. (a) A visualization of the XOR problem in 2D space. A single linear classifier in 2D space fails to perform the XOR problem. (b) If the response variable is projected into 3D space, this classification problem is achievable using a single linear classifier

A similar behavior is realized in RC. The connection of various nonlinear nodes (neurons) – called the Reservoir – results in a high dimensional project. The results of the RC are then simply extracted via a weighted sum of the neuronal state of the reservoir neurons using a linear classifier called the

readout circuit. Interestingly, since the connection weights within the RC are randomized, they require little to no training beyond an initial parameter space searching. Instead, training RC is simply performed via training the readout circuit. If a linear readout circuit is chosen, then a simple linear regression, or a ridge regression [80], following (3.4)

$$W = \left(X^T X + k_{reg} I\right)^{-1} (X^{-1} Y) \tag{3.4}$$

Where W is the readout weight matrix, X is the neuronal state matrix, Y is the expected output of the RC, k_{reg} is a small regularization constant and I is the identity matrix. The sizes of W, X, Y and I are $N \times R$, $M \times N$, $M \times R$, and $N \times N$, respectively. Where N is the number of neurons in the reservoir, M is the number of training samples, and R is the number of reservoir outputs. The output of the RC, S, is then written as:

$$S = XW \tag{3.5}$$

Multiple readout circuits can be connected to the same reservoir simultaneously, allowing the RC to perform parallel computing seamlessly. In fact, an ideal RC is theoretically capable of performing universal approximation [81].

For a reservoir computing scheme to be successful, the reservoir should satisfy three requirements:

Three requirements were loosely defined in the literature for a reservoir dynamical system:

R1. Input separability

R2. Possessing fading memory

R3. Echo state property

R1 signifies the system's ability to differentiate between two different inputs by mapping them to distinct outputs. This requirement is typically satisfied when the system possesses sufficient nonlinear complexity. While R2 and R3 tend to be correlated in most systems. These properties

represent the ability of the system to prioritize recent inputs and forget past inputs and eventually forget past inputs. The RC requirements are loosely defined in the literature. However, most works agree on the following requirements:

For R1 to be satisfied, when a reservoir computing scheme $\psi(.)$ is subjected to two distinct input streams for the reservoir computer $u_1(t)$ and $u_2(t)$, the reservoir output is Y_1 and Y_2 such that $Y_1 \neq Y_2$. This requirement is satisfied in the chaotic regime, where the system separates even extremely similar inputs. However, in practice, the separation property should take in consideration that similar inputs are still mapped to the same output to eliminate sensitivity to noise. Thus, operation in the chaotic regime is not ideal.

Properties R2 and R3 are satisfied if the reservoir state is only a function of a finite number of past input values, up to τ_m. Mathematically, this can be written as (3.6):

$$x(t) = F\big(x(t-1), u(t-1), \dots, u(-\infty)\big) = F\big(x(t-1), u(t-1), \dots, u(t-\tau_m)\big) \qquad (3.6)$$

A particular example that satisfies R2 and R3 is operation around a fixed-point attractor. In this case, $x(t) = F\big(x(t-1)\big) = x(t-1)$. However, here, the system exhibits simple dynamics, which makes it ill-suited for the separation property. Interestingly, there exists a region that offers great separation while retaining the echo state property. This region, named the edge of chaos, was previously found in Boolean RNNs applications to be ideal for computation [79] and later in reservoir computing [82]. In fact, there exists a hard edge between chaos in a lot of computational systems, at which computation is preferred. Bifurcation diagrams (Section 2.3) can be of great benefit to identify this region.

In practice, however, a perfect RC is difficult to attain. Additionally, in their traditional form, physical reservoirs may be impractical to employ in hardware due to the need for hundreds or thousands of nonlinear nodes. This problem was recently solved through the development of delay-based reservoir computing [39].

In delay-based RC, physical nodes are replaced with virtual by utilizing a single nonlinear dynamical system. By definition, the state of a dynamical system is functions of their previous time states. Therefore, one can view such dynamical system as a network of serially connected neurons. If a delayed feedback loop is introduced, the system will then resemble a network of recurrent neurons instead. In this dissertation, the nonlinear dynamical system chosen is a MEMS accelerometer.

It is noted here that reaching an attracting fixed point or a periodic orbit is analogous to neurons decoupling as the dynamical system state is bound to a fixed orbit. To avoid this, the system must retain transience. To this end, input modulation has been proposed as a means to force transience and enable RC [39]. In systems in which sensing and computing and decoupled, delay-based RC is performed. The input signal $u(t)$ is sampled-and-held at a frequency of $1/\tau$ to generate the signal $I(t)$ following (3.7)

$$I(t) = u(i\tau),\ i\tau < t < (i+1)\tau,\ i = 0,1,2,\dots,T-1 \tag{3.7}$$

Where T is the total number of input samples captured by the sample-and-hold circuit. To modulate this signal, a periodic mask M is applied following (3.8)

$$J(t) = I(t)m(j),\ (j-1)\theta + i\tau \le t < (j)\theta + i\tau,\ j = 1,\dots,N-1 \tag{3.8}$$

Where $J(t)$ is the modulated signal, $m(j)$ is the discrete periodic mask, $\theta = \tau/N$ is the duration of each mask value and N is the number of virtual neurons. The mask is chosen to be periodic to facilitate processing without losing information, as aperiodic masking is simply a single distortion. The random mask values are often chosen to be binary. However, tertiary and more complex masks may also be used. As transience is important in delay-based RC, the choice of random mask appear to have a substantial effect on the network response and the behavior of the virtual reservoir.

Delayed-feedback may be introduced by utilizing the states of the dynamical system. While negative delayed-feedback was found to be useful to stabilize some nonlinear systems like MEMS

devices [83], the goal here is to introduce further nonlinearities and create recurrent connections. Thus, positive delayed feedback with a delay duration of τ and amplitude of α is used instead here.

The modulated signal is fed to the dynamical system, which changes its state. The states of the dynamical signal are collected at θ intervals as the states of the virtual neurons. In practice, these states are collected at intervals of $\theta + \delta$, where δ is a small delay, to enable the dynamical system to react to $J(t)$ by accounting for its inertia. The neuronal state matrix is then written as

$$X_{ij} = x(j\theta + i\tau),\ i = 0,1,\dots,T-1,\ j = 0,1,\dots,N-1 \quad (3.9)$$

It should be noted here that, unlike traditional RC schemes, delay-based RC generates neuronal outputs serially. This means that this scheme can only perform computing at τ intervals once all virtual neuron states have updated. Thus, there exists a tradeoff between the complexity of the reservoir, relating to the number of virtual neurons ($N = \tau/\theta$) and the computational speed.

Training is performed offline using linear regression, if a linear readout circuit is used. Other types of readout circuits may be used to incorporate nonlinearities into the readout rather than the reservoir [84] or to improve the performance of the RC scheme. However, in this thesis, linear readout circuits are strictly used to ensure most computation is performed at the sensor level.

CHAPTER 4

MEMS DYNAMICS FOR COMPUTING

Computation near the sensor node has been shown to solve some common problems in smart systems, such as latency. This process is named Edge Computing [14]. In this chapter, a neuromorphic computing scheme is demonstrated at the sensor level by utilizing a network of MEMS devices as a CTRNN. This approach is possible due to the nonlinear behavior of MEMS devices and their bistability in some operational regimes.

In this chapter, two MEMS structures are considered, a SDOF electrostatically actuated straight MEMS device and a SDOF electrostatically actuated MEMS arch. Unless otherwise noted, the parameters of each MEMS device are given by tables 2.1 and 4.1, respectively.

Table 4.1: MEMS arch parameters used in this chapter.

Parameter	Value
L	$1\ mm$
b	$30\ \mu m$
d	$10.1\ \mu m$
h	$3\ \mu m$
ρ	$2330\ kg/m^2$
E	$160\ GPa$

4.1. MEMS Dynamics Modification

For a network of MEMS devices to be used as a hardware alternative to a CTRNN, the dynamics of a single MEMS device must be qualitatively similar to that of a single continuous-time recurrent neuron (CTRN). To this end, some critical behaviors of CTRNs and small CTRNNs will be

presented below with the aim to be duplicated by individual MEMS devices and small MEMS networks, respectively.

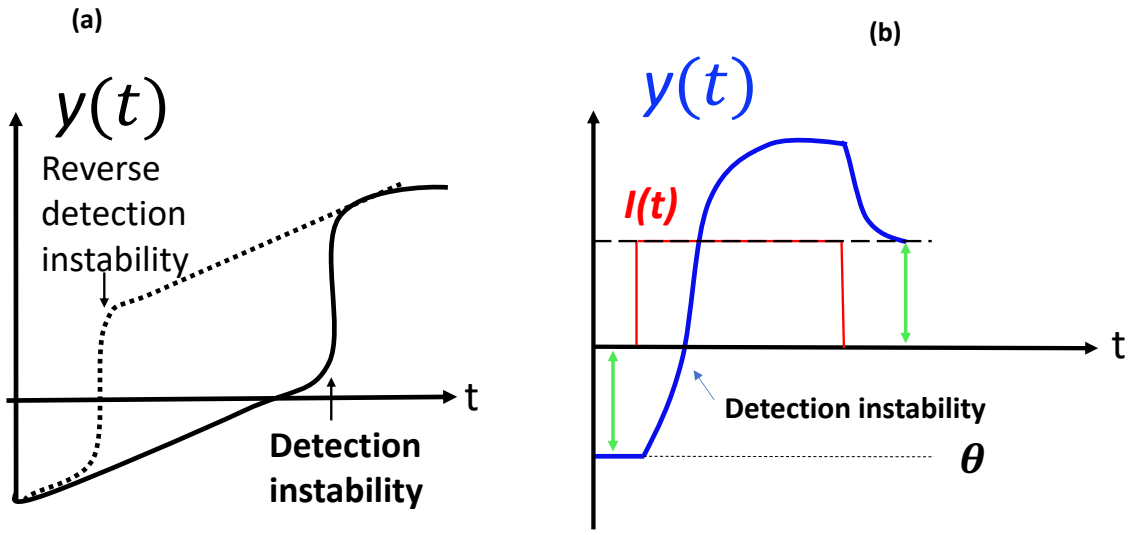


Figure 4.1. Basic behaviors of a single CTRN. (a) Detection: a ramp input signal with a positive slope is applied resulting in an increasing CTRN response (solid line). Beyond some threshold value, a sudden response jump is noted, named "detection instability". When a negative-sloped ramp signal is applied instead, the response of the CTRN decreases steadily (dashed line). A sudden response jump is observed below some threshold, named "reverse detection instability". Note that the reverse detection instability threshold is lower than the detection instability threshold. (b)Memory: a rectangular input is applied to a CTRN. Prior to the introduction of the input, the CTRN state is negative. Once the input is applied, detection instability is noted. Finally, after the input signal dies out, the CTRN state decreases. However, it does not experience reverse detection instability. Thus, the final state of the CTRN is larger than the initial state (green lines).

It is noted here that continuous time recurrent neurons (CTRNs) and Dynamic Field Theory (DFT) [32] neuronal populations are discussed interchangeably due to the mathematical similarities of both models. Detection, memory and selection are described as the building blocks of neuronal population behaviors. The former behaviors are inherent to individual neuronal populations, and thus, CTRNs, while the later behavior is exhibited in small networks.

4.1.1. Dynamics of a single CTRN

A single CTRN, or alternatively, a single neuronal population, should exhibit both detection and memory. Detection in the context of neuronal populations refers to a sudden response jump due to the introduction of a sufficiently large input signal (detection instability) or due to the removal of an input signal (reverse detection instability), as shown in FIG.4.1. Psychologically, detection

explains how a certain sensory input, received by the human central nervous system, can trigger an event (such as a motor action) once it exceeds a threshold value. Memory refers to the ability of a CTRN to maintain its state once an input is removed. Psychologically, it explains how a memory neuron can maintain the effects of strong stimulus after it is no longer present in order to decide future actions.

For both detection and memory properties to be satisfied, detection instability and the reverse detection instability must occur at differing input levels, I_D and I_R, respectively. Therefore, there are some input levels $I_R < I < I_D$ such that the DFT neuronal population response can take one of two values, depending on whether the input is increasing or decreasing, thus, depending on the initial condition. Dynamically, this response is named Bistability. Hysteresis, which exists in bistable regimes, is of profound importance for computation.

Three well-known MEMS operational regimes exhibiting hysteresis:

1. The pull-in regime in electrostatic MEMS devices.
2. The snapthrough regime of MEMS arches.
3. The bistable regime in low-parasitic capacitance, electrical-resonance driven electrostatic MEMS.

4.1.1.1. The pull-in regime in electrostatic MEMS devices

The bistable response in a MEMS CTRN arises from the coexistence of two stable response fixed points simultaneously. Strictly speaking, pull-in is an instability rather than a bistability. However, as it exists in any simple attracting electrostatic MEMS structures, it is discussed here nonetheless. The difference between instability and bistability is highlighted in FIG.4.2 and FIG.4.3. The bifurcation diagram of a DFT neuronal population (FIG.4.2,a) shows a double-saddle-node bifurcation, which separates the potential well of the DFT neuronal population response into two wells (not shown here). Each well is stable, resulting in bistability. The system can jump from one

stability well to another if sufficient input signal is applied. Moreover, the response of the neuronal population depends on the input amplitude. At low input amplitudes, only a single (stable solution exists). At higher input strengths, 3 solutions exist: two stable solutions (black points) and one unstable solution (grey point) separating them (FIG.4.3,b).

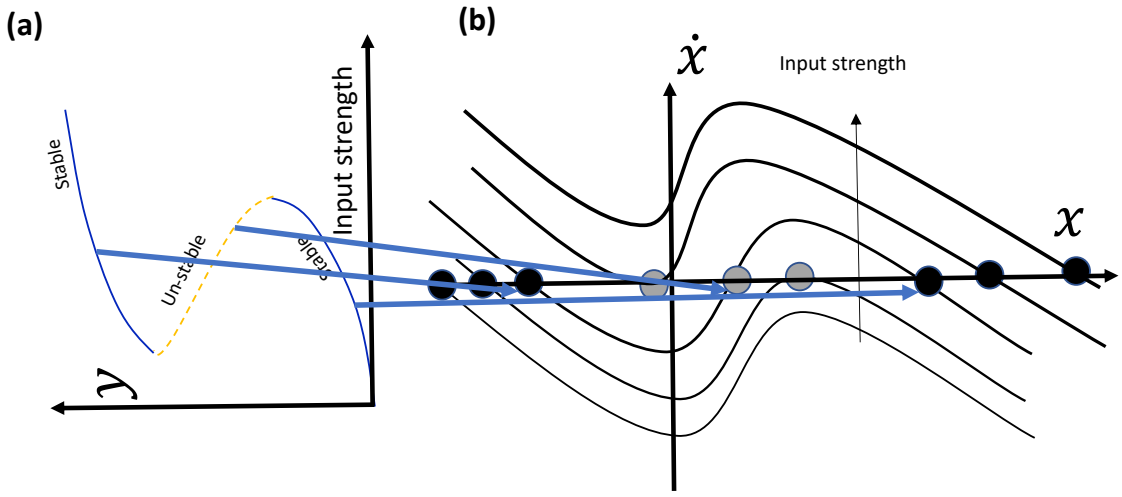


Figure 4.2. Bistability exhibited in a DFT neuronal population (similar to a CTRN). (a) Bifurcation diagram of the DFT neuronal population showing a double saddle-node bifurcation with respect to the input strength (amplitude of $I(t)$). This bifurcation enables the coexistence of two stable solutions (solid lines). (b) Corresponding response phase portrait showing the existence of 3 solutions: two stable and one unstable, corresponding to the x-axis intersection. We note here that some input strengths only allow for a single stable solution.

Contrary to the DFT population, the MEMS device experiences a single saddle-node bifurcation (FIG.4.3,a) at pull-in. Thus, only one potential well exists in this case and bistability is not attained. The phase portrait of the MEMS device (FIG.4.3,b) shows this case holds for any input (voltage) value. Despite this observation, however, hysteresis is still exhibited for a MEMS device in the pull-in regime, as shown in FIG.4.4. In fact, the pull-in instability corresponds to a rapid response jump similar to detection instability (FIG.4.4,a). Hysteresis shown in FIG.4.4,b ensures that reverse detection instability (following the red line) is lower than the detection instability (blue line), thus enabling memory as well. Therefore, even in their basic behavior, a single electrostatic MEMS device, operated around pull-in, exhibits both detection and memory. While this approach may be appropriate for some applications, other applications may be hindered due to the response cap at

pull-in, due to electrode contact. Additional concerns of utilizing this method include stiction and surface wear due to frequent operation around the pull-in regime.

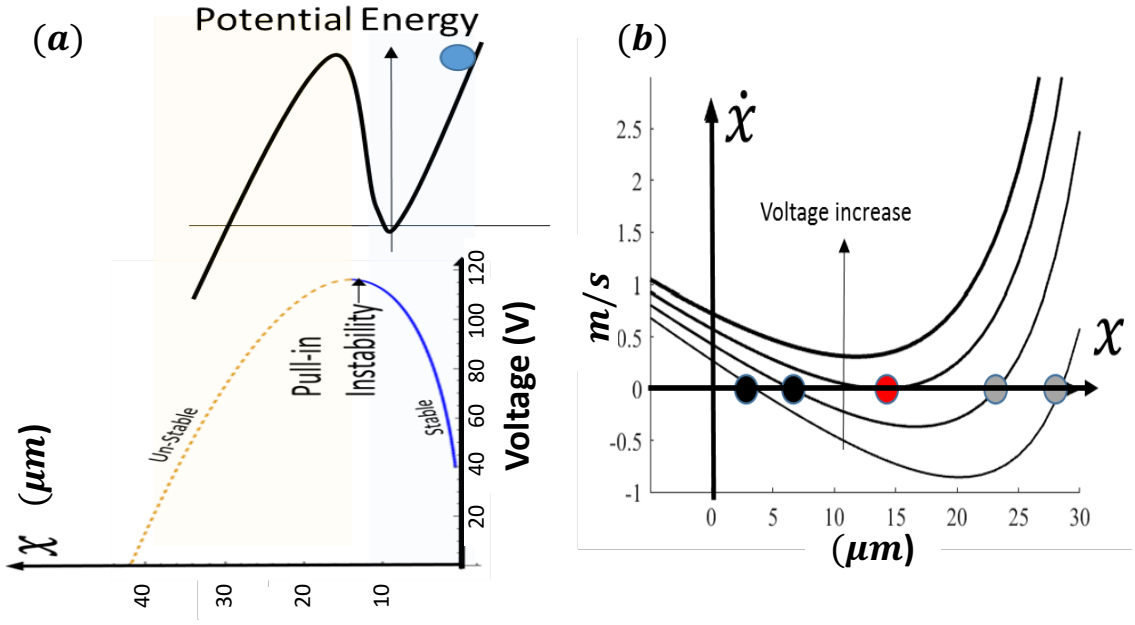


Figure 4.3. Response of a simple SDOF electrostatic MEMS device. (a) Potential energy of the MEMS device and bifurcation diagram (as a function of voltage) showing a single saddle-node bifurcation occurring at pull-in, after which, instability ensues. It is noted here that the stable solution is represented by a well in the potential energy plot. (b) Phase portrait of MEMS response showing the existence of only two solutions. A stable solution (low amplitude, black dot) and an unstable solution (high amplitude, grey dot). The onset of pull-in is shown as a red point.

4.1.1.2. The snapthrough regime of MEMS arches

The potential problems of utilizing MEMS in the pull-in regime can be solved when hysteresis is achieved prior to contact. A commonly analyzed regime of instability is the snapthrough regime in MEMS arches. The response of a SDOF MEMS arch are described using (4.1) [74]

$$\frac{3\rho LA_{cs}}{8}\ddot{x}(t)+\frac{3Lc_{eff}}{8}\dot{x}(t)+\frac{2\pi^4 EIc}{L^3}(x(t)+b_0)+\frac{\pi^4 EIc}{8L^3}(x^2(t)-b_0^2)x(t)=\frac{\varepsilon A_s V_{MEMS}^2}{4\sqrt{d(d-x(t))^3}} \quad (4.1)$$

Where ρ is the mass density of the microbeam, L is the length of the microbeam, A_{cs} is the cross sectional area, x is the deflection from the equilibrium point (with no force, positive in the direction away from the substrate) and the dot operator represents temporal derivatives, c_{eff} is the damping coefficient, E is the Young modulus of elasticity, I_c is the second moment of area of a straight

microbeam with the same dimensions about the microbeam's neutral axis, b_0 is the initial midpoint elevation, and A_s is the surface area.

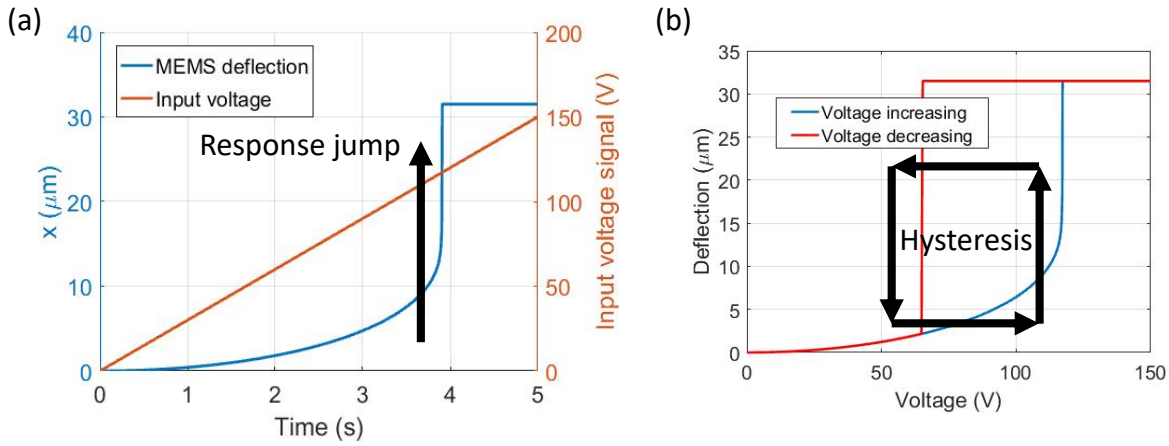


Figure 4.4. MEMS response around pull-in showing both a response jump (a) and hysteresis (b).

Assuming high damping, the first term in (4.1) can be dropped. Equation (4.1) can be further simplified by defining the variables in Table 4.2 to resemble a nonlinear ODE with a cubic nonlinearity in the form:

$$\tilde{\tau}\dot{x}(t) + \tilde{k}_1 x(t) + \tilde{k}_3 x^3(t) = \tilde{C} + \tilde{f}(x) \tag{4.2}$$

Equation (4.2) can consequently be written in a form similar to a CTRNN equation (3.3):

$$\tau\dot{x}(t) = -x(t) + \theta + w_{11}\sigma_{Arch}(x) + I_{MEMS}(x, V_0 + V_a) \tag{4.3}$$

where the constants and the parameters are also defined in Table 4.2. We note that the feedback function of an arched microbeam $w_{11}\sigma_{Arch}(x)$ also exists in a straight clamped-clamped beam due to mid-plane stretching, where w_{11} is the ratio between the cubic nonlinear stiffness k_3 to the linear stiffness k_1. However, in a straight beam k_1 far exceeds k_3 by orders of magnitude, leading to a very small gain w_{11}. In contrast, the initial curvature of an arch microbeam reduces the linear stiffness to an effective linear stiffness of $k_1 = k^* - b_0^2 k_3$. Thus, as the initial curvature increases, k_1 decreases until it reaches the same order of magnitude of k_3. Once k_3 becomes appropriately

larger than k_1, snapthrough is observed and the bistability behavior, similar to the DFT neuronal population, is observed. This behavior is noted in FIG.4.5. Bistability is observed when $b_0 = -4\ \mu m$ and $b_0 = -3\ \mu m$ while $b_0 = -2\ \mu m$ results in no bistability.

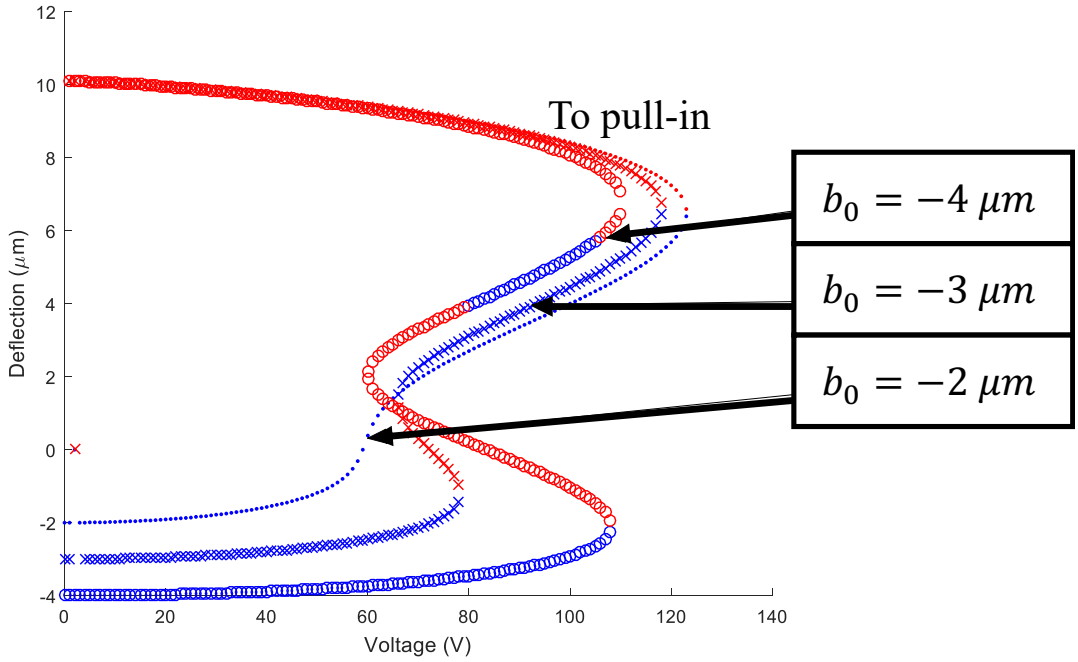


Figure 4.5 Bifurcation diagram of a MEMS arch with various mid-point elevations. Bistability is observed when $b_0 = -3\mu m$ and $b_0 = -4\mu m$ while it is absent when $b_0 = -2\mu m$.

Table 4.2: Parameters of equation 4.3

Parameter	Definition
$\alpha = \frac{c_{eff}}{\rho A_{cs}}$	Damping parameter
$\omega_{m1} = 22.3733\sqrt{\frac{EI}{\rho A_{cs} L^4}}$	First modal frequency of a straight clamped-clamped beam
$k^* = \frac{16\pi^4}{1502}\left(\frac{\omega_{m1}^2}{\alpha^2}\right)$	Linear stiffness of a straight clamped-clamped beam

$k_1 = k^* - \left(\frac{b_0^2}{d^2}\right) k_3$	Linear stiffness of an arched beam
$k_3 = 0.065 d^2 \frac{A_{cs}}{I} \frac{\omega_{m1}^2}{k_1}$	Cubic stiffness
$\tau = \frac{\alpha}{k_1}$	Time constant
$w_{11} = -\frac{k_3}{k_1}$	Self-feedback
$\theta = \frac{16\pi^4}{1502} \frac{b_0 \omega_{m1}^2}{\alpha^2 d\ k_1}$	Offset
$I_{MEMS}(x, V_{MEMS})$	Input (excitation) signal
$= \frac{\Gamma(x(t), V_{MEMS})}{k_1}$	
$\sigma_{Arch} = x^3$	Feedback function

As an example of MEMS arch as a neuronal element, a MEMS arch with parameters given in Table 4.1 is used. To enable bistability, the initial midpoint elevation is chosen to be $b_0 = 3\ \mu m$. These dimensions are commonly used in the literature for MEMS arches [74]. The response of this MEMS device is shown in FIG.4.6. In this figure, detection instability occurs when the input voltage exceeds the snapthrough (jumping from the low deflection region to the high deflection region) voltage, or the detection voltage V_D

and reverse detection instability occurs when the input voltage is below the release voltage, or reverse detection voltage V_R. This figure resembles the neuronal response in FIG.4.1. The bi-stability and hysteresis in the arched beam can also be used to simulate a memory behavior. In this case, the MEMS is initially biased with a bias voltage. Once the MEMS is excited by an additional input signal (a step function in this case), it drives the MEMS to snap-through, achieving high deflection. Once the input signal is removed, the MEMS remains on the top branch of the bifurcation curve if the bias voltage, V_0, satisfies $V_0 > V_R$ (FIG.4.6,b).

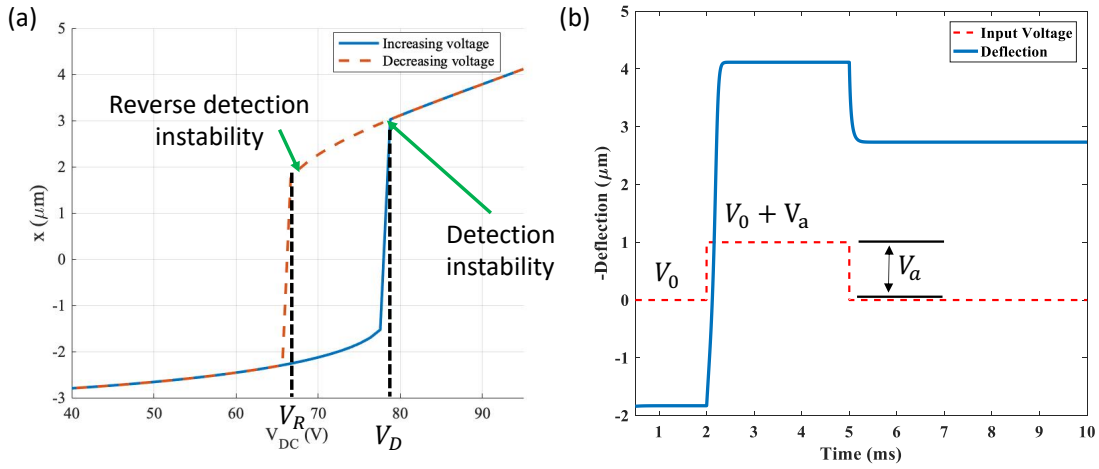


Figure 4.6 (a) Bifurcation diagram of a single MEMS arch showing the detection process. Hysteresis is also shown in this regime via the snapthrough process. Note the lack of amplitude capping in this case. (b) Memory behavior of the MEMS arch. The MEMS device is biased using $V_0 = 75V$. The MEMS is later excited using a step signal with an amplitude $V_a = 20\ V$.

4.1.1.3. The Bistable Regime in Low-Parasitic Capacitance, Electrical-Resonance Driven Electrostatic MEMS

Bistability can also be introduced to straight MEMS devices by utilizing self-feedback. This can be achieved by utilizing electrical resonance to drive the MEMS device. As ideal MEMS devices are modeled as variable capacitors, an external inductor is added to the circuit to create an electrical resonant (LC circuit). Additionally, a resistor is added externally to limit the current in the circuit. An ideal MEMS RLC circuit is shown in FIG.4.7,a. while a realistic model of the MEMS circuit is shown in FIG.4.7,b. This model is based on the experimental design, shown in chapter 7. Parasitic capacitances C_{BNC1} and C_{BNC2}, and parasitic resistances R_{BNC1} and R_{BNC2} are introduced due to the use of coaxial, BNC, cables to interface with the inputs and outputs of the circuit. The MEMS device itself is modeled in the middle branch having a parallel resistor R_{PP} and capacitor C_P to the MEMS variable capacitance C_{MEMS}. An additional serial parasitic resistance R_{PS} is also present in the realistic model. Additional explanation about the dynamics of the MEMS device within an LC circuit is also found in chapter 7.

Assuming the MEMS is actuated using an AC source only with a frequency $\Omega \gg \omega_m$, which is usually the case when operating about the electrical resonance frequency, the forcing function can be written as

$$F_e = \frac{\varepsilon A_s (V_{AC}\cos(\Omega t))^2}{2\left(d - x(t)\right)^2} \tag{4.4}$$

Expanding the squared term using trigonometry:

$$(V_{AC}\cos(\Omega t))^2 = V_{AC}^2\left[\frac{1}{2} + \frac{1}{2}\cos(2\Omega t)\right] = \frac{V_{AC}^2}{2} + V_{Ac}^2\cos(2\Omega t) \tag{4.5}$$

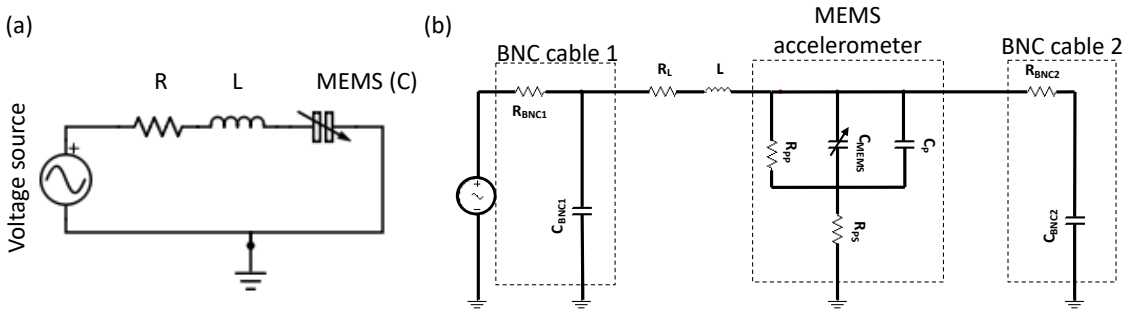


Figure 4.7. Electrical model of the MEMS device. (a) Ideal MEMS RLC circuit, excluding all the system and source parasitic. (b) Schematics of the MEMS circuit including the parasitic capacitances (C_{BNC1}, C_{BNC2}, and C_p) and parasitic resistances (R_L, R_{pp}). The characteristic impedances of the cables (R_{BNC1}, R_{BNC2}) are insignificant at low frequency and are ignored in the circuit. While R_S is the output resistance of the source, R_{IN} is the input impedance of the oscilloscope, which is significantly large in magnitude. The output voltage is read across C_{BNC2}.

Linearizing the system, such as the output of the MEMS equals the superposition of the output of the two terms in (b). The MEMS will attenuate the second term (with a frequency much higher than the resonance frequency) while viewing the first term as a DC input equivalent to:

$$V_{DCequivalent} = \frac{V_{AC}}{\sqrt{2}} = V_{RMS} \tag{4.6}$$

Electrical resonance amplifies $V_{DCequivalent}$. Thus, $V_{MEMS,}$ can be approximated by:

$$V_{MEMS} = \beta(x)\frac{V_{AC}}{\sqrt{2}} = \beta(x)V_{RMS} \tag{4.7}$$

where V_{RMS} is the root-mean-square of the AC signal and $\beta(x)$ is the electrical gain; assuming a simple RLC circuit, is given by:

$$\beta(x) = \frac{1}{\sqrt{\left[\left(1-\omega^2 LC(x)\right)^2+\left(\omega RC(x)\right)^2\right]}} \tag{4.8}$$

where $C(x)$ is the equivalent capacitance of the MEMS, BNC cable, and peristatic capacitances. Thus, the voltage amplification is a function of the MEMS deflection that induces a negative feedback. This is true as the MEMS deflection increases due to voltage amplification, the capacitance of the MEMS resonator changes, which shifts the electrical resonance frequency, attenuating the input voltage signal, changing the electrical resonance again, and the cycle keeps repeating until the MEMS reaches its final stable attractor. By activating this passive feedback, and by assuming the MEMS is to run in ambient air, relaxing the need for the expensive vacuum packaging requirement for a typical MEMS [85], where the inertial term can be ignored, the MEMS dynamics equation transformed to a form that resembles the neuronal population dynamics equation as shown in (4.9), assuming sufficiently high damping:

$$\tau\dot{x}(t) = -x(t) + \Gamma_{\mathrm{e}}[V_0 + V(t) + \beta(x)V_{RMS}, x(t)] \tag{4.9}$$

Where $\Gamma_e[.]$ is a nonlinear kernel due to electrostatic actuation around the electrical resonance, given by (4.10):

$$\Gamma_e\big(V, x(t)\big) = \frac{\varepsilon A_s (V)^2}{2(d-x(t))} \tag{4.10}$$

The resemblance between neuronal dynamics and (4.9) are confirmed in FIG.4.8, where bifurcation diagrams and phase portraits of the MEMS are plotted. FIG.4.8,a shows an ideal comparison (assuming very low parasitic capacitance) between the response of the MEMS to a normal DC input exhibiting a saddle-node bifurcation at $V_{MEMS} \approx 120\ V$ and when the MEMS is actuated around the electrical resonance, where the MEMS can avoid pull-in almost entirely and experience a bi-stable dynamic. FIG.4.8,b shows the phase portrait of the MEMS dynamic with electrical resonance

activation. The figure confirms that by activating electrical resonance, the MEMS dynamic is transformed to resemble the neuron dynamic. Next, this new transformed MEMS dynamic are utilized to simulate the two basic operations of a single CTRN neuron; detection and memory. FIG.4.9,a represents the use of the MEMS device as a detection neuron by utilizing the bi-stability jump of the MEMS at some critical voltage (V_D). As the voltage increases, the MEMS microbeam steadily deflects towards the stationary electrode until $V = V_D$, where the MEMS exhibits a sudden jump, changing its fixed (equilibrium) point to one closer to the electrode (forward sweep – Blue solid line). This represents the detection instability. Later, when the voltage is reduced slowly, the MEMS steadily deflects away from the stationary electrode and remains this way past $V = V_D$, exhibiting hysteresis. The MEMS rapidly converges to the lower branch when $V = V_R$ (backward sweep – Red solid line); representing the reverse detection instability.

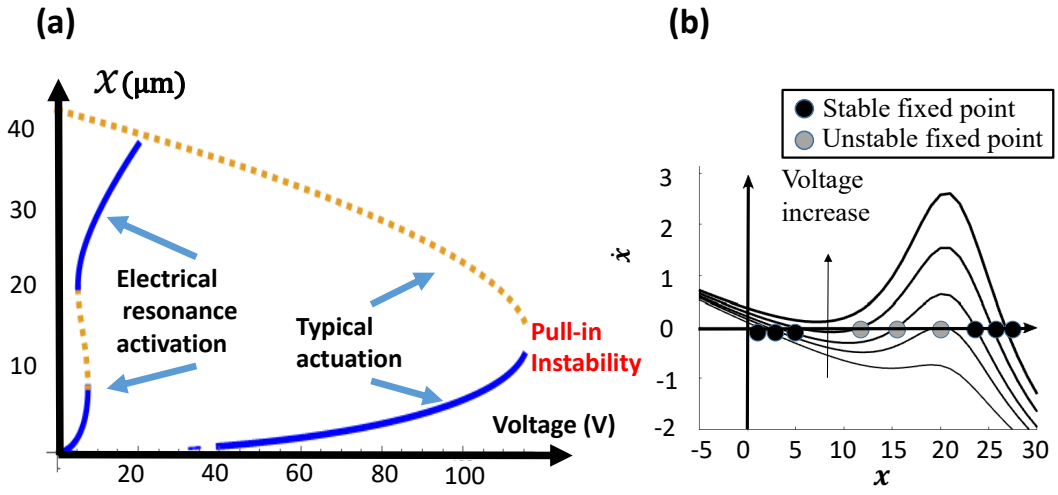


Figure 4.8 The MEMS dynamics including the effect of electrical resonance. (a) Bifurcation diagrams comparing the MEMS dynamics with and without electrical resonance activation. (b) Phase portrait with electrical resonance activation.

FIG.4.9,b shows the response of the MEMS device as a memory by maintaining a high amplitude position even when the input voltage, V_{in}, is removed. The MEMS is initially biased with a combination of AC and DC signals (V_0). V_0 is chosen such that it exceeds the reverse detection critical voltage V_R. The MEMS exhibits low deflection initially. Then, it is excited by an AC signal

such that ($V_{in} + V_0 > V_D$) which triggers the bi-stable jump of the microbeam to a high amplitude position. The excitation signal is then removed, but the MEMS stays on the upper branch of the bifurcation diagram (Fig.10a) since ($V_0 > V_R$). Thus, the microbeam retains "memory" of the input signal. This memory is retained until it is cleared by having $V_0 < V_R$ which leads the MEMS into the lower branch of the bifurcation diagram.

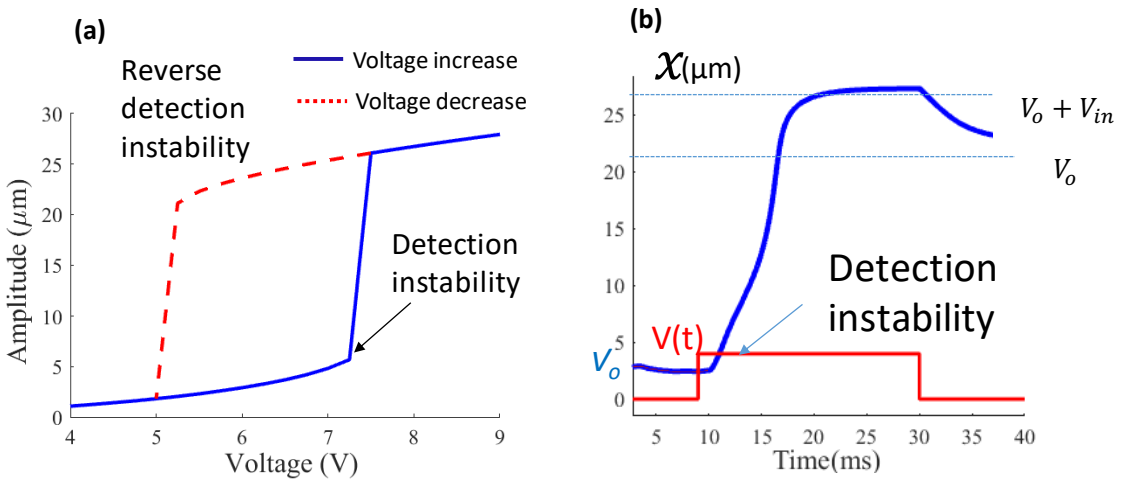


Figure 4.9 Theoretical investigation of a MEMS behavior as a neuron. (a) Detection neuron by producing high amplitude jump when activated, through detection instability. (b) Memory neuron by retaining high amplitude position after the excitation signal is lost (remembering the excitation signal).

4.1.2. Dynamics of a small CTRNN

Selection is a building block of neuronal populations according to DFT. This behavior allows neuronal populations to favor a certain event over another. The coupling between multiple connected neurons, resulting in multiple local peaks, can be used to explain more advanced behaviors such as the selection process [32], in which global inhibition maintains one peak due to a certain sensory input while suppressing others. This inhibition behavior can be realized for two interacting neuronal populations (FIG.4.10) by having negative coupling strength for w_{11} and w_{12} in (3.3) describing their coupled dynamics.

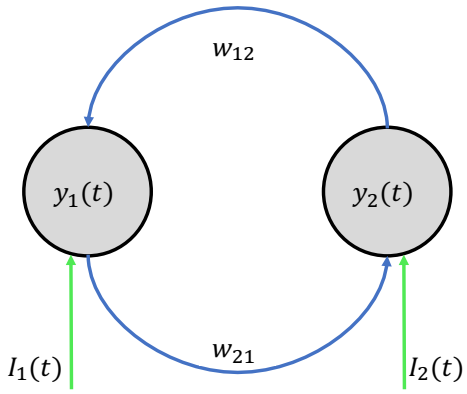


Figure 4.10 The dynamics of two neuronal populations. Arrow indicates excitatory input and circle indicates inhibitory coupling.

The coupling between the two MEMS neurons is shown in Fig.4.11, where the following variables are defined: $V_1 = (V_{in})_1 + \mathrm{V_{o1}} - w_{12}V_{Out2}$,$V_2 = (V_{in})_2 + \mathrm{V_{o2}} - w_{21}V_{Out1}$, $V_{Out1} = V_1\mathrm{H_1}(x)$, $V_{Out2} = V_2\mathrm{H_2}(x)$. V_i is the output voltage of the i^{th} OP Amp, $(V_{in})_1$, V_{oi} are the input and bias voltages of the i^{th} MEMS, V_{Outi} is the output voltage of the i^{th} MEMS w_{ij} is the gain of the connection from the j^{th} MEMS to the i^{th} MEMS, generated through an OP Amp, and H_i(x) is a transformation function between deflection and voltage for the i^{th} MEMS. The above gains can be tuned through the resistors and will signify the magnitudes of inhibition and amplification in the circuit. A similar arrangement was used to couple memristive devices for computing purposes. In this circuit, operational amplifiers are used to sum/subtract the voltages and isolate the MEMS neuron circuits from one another. Passive mechanical coupling between MEMS devices is also possible by introducing multiple microbeams sandwiched between the excitation electrodes.

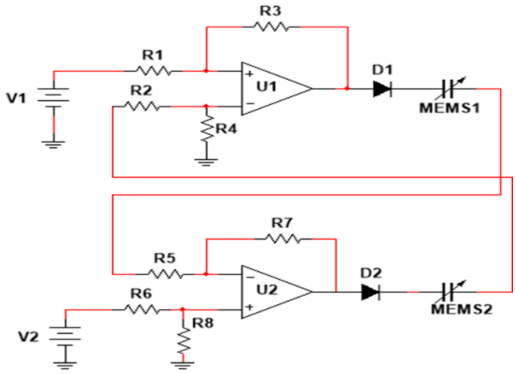


Figure 4.11 Electrical connection between two MEMS devices. In such connection, the voltage signal across one MEMS is used to inhibit the response of the other MEMS through the differential amplifier.

We note that this design can only be achieved because the external negative coupled term is a function of the MEMS displacement. Thus, the voltage across the MEMS offers a feedback signal that may be used to inhibit the other MEMS device producing a selection mechanism.

Figure 4.12 shows an example of the selection process using two arched-beam MEMS neurons. In FIG.4.12,a, we show their dynamics response. In the figure, any MEMS deflection exceeding 3μm is a snap-through response that will close/short the MEMS circuit with a stopper that is connected to the MEMS substrate. FIG.4.12,b shows the input voltage for both neurons. The figures show that initially both neurons are biased by 20 V and 24 V, respectively. By itself, the bias voltage is not enough to trigger snap-through for any of the MEMS. However, at 1 ms, a 90 V was added to the MEMS #1 resulting in snap-through response and closing the MEMS circuit. Thus, the second MEMS neuron is subjected to a negative, inhibitory, voltage equals to $V_1 H_1(x)$. So even later at 3 ms, 80 V was added to the second MEMS, the second MEMS stayed in the OFF state. However, when the input voltage at the first MEMS dropped to its bias voltage, the second MEMS exhibited snapthrough and closed its circuit.

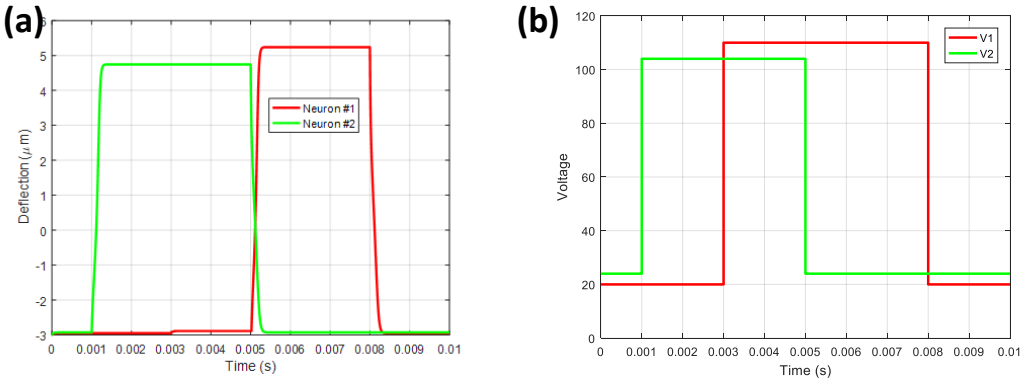


Figure 4.12 The response of two MEMS neurons demonstrating the selection process. (a) MEMS deflections, representing the state of the CTRN neurons. (b) Input voltages to each MEMS device.

Other interesting dynamics can arise from the interactions between two MEMS devices, such as the emergence of periodic orbits [78] as shown in Fig.4.13. In this figure, the two MEMS devices

are biased by $V_{o1} = V_{o2} = 20V$ and the second MEMS neuron is excited by an input voltage $(Vin)_2 = 70V$, while the first MEMS is not excited. To achieve the periodic response, we set $w_{12} > 0$ and $w_{21} < 0$. In this case, when the Neuron #2 snaps-through due to $(Vin)_2$, the circuit is closed and the output voltage excites the first MEMS neuron because $w_{12} > 0$. However, when the first MEMS snapsthrough, this inhibits the second neuron because $w_{21} < 0$, thus releasing the second MEMS. However, this release would also cut off the voltage supply to the first MEMS, thus allowing the second MEMS to snap-through again. The continuous repetition of this process produces the period orbit of Fig.4.13.

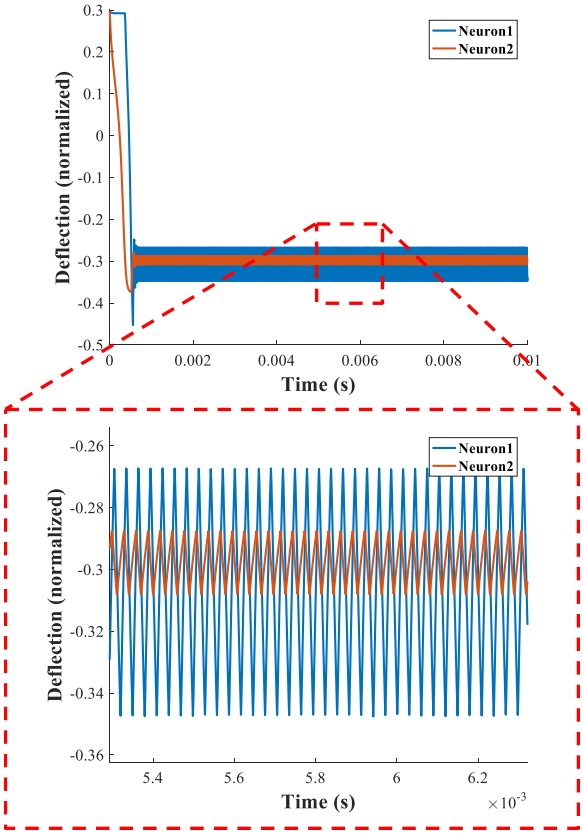


Figure 4.13 The response of two MEMS neurons demonstrating the emergence of a periodic orbit in the system when a DC supply is applied to MEMS#2. Top: Full view. Bottom: Zoomed view.

4.2. MEMS CTRNN Experimental Analysis

To demonstrate dynamical changes of a MEMS network response experimentally, two identical, electrostatically-actuated double-cantilever MEMS accelerometers were used. Each of the MEMS microbeam has dimension given in Table 2.1. Despite its large in-plane dimensions, this device retains the same qualitative behaviors as smaller MEMS devices [73], including hysteresis and bistability, which are essential for neuro-inspired computing. The two MEMS devices were coupled in a small network as shown in Fig.4.14. A data acquisition device was used to record the voltage across the MEMS devices and to produce the required coupling. Due to the slow response time of the MEMS devices used here (a fundamental resonance frequency of 190 Hz), the effects of analog-to-digital convergence (A/DC) and digital-to-analog convergence (D/AC) delays were negligible. Alternatively, operational amplifiers in series can be used to couple the MEMS devices in the network to bypass these delays and operate the system in a purely analog fashion. In this work, programmatic coupling was used as it offers more flexibility in choosing coupling weights. The effective voltage across a MEMS device in the constructed network is given by (4.11):

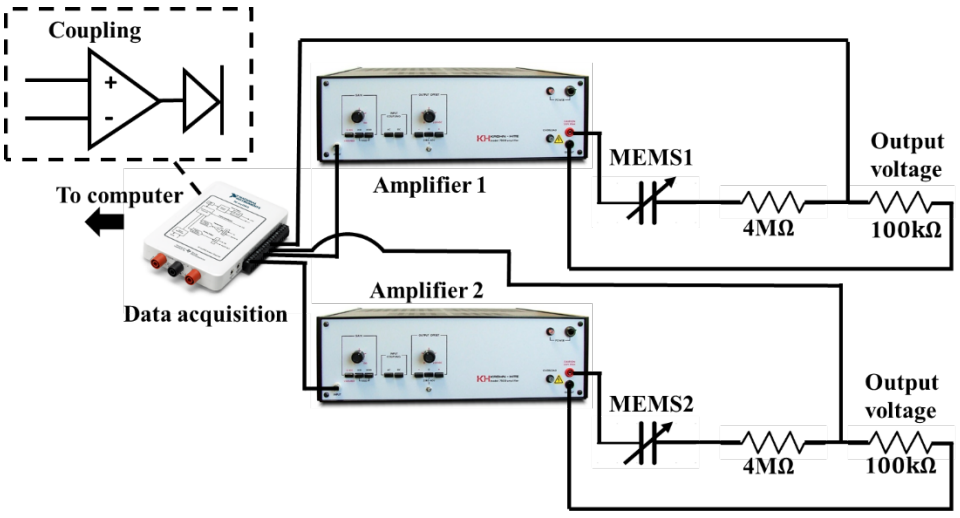


Figure 4.14 The network connection circuit. The two MEMS devices are actuated by a signal from the data acquisition device that is externally amplified. Each MEMS is connected to a 4 MΩ resistor, to reduce the current flowing at pull in, and a 100 kΩ for voltage division.

$$V_{MEMS_i} = V_{in_i} + w_{ij} V_{out_j} \tag{4.11}$$

When the MEMS devices are uncoupled ($w_{12} = w_{21} = 0$) and the electrostatic force acting upon either of them exceeds their stiffness force, the microbeam collapses (pulls-in) as shown in Fig.4.15(a). Hysteresis and bistability are evident in the figure between the pull-in voltage ($V_{pull\text{-}in}$) and the pull-out voltage ($V_{pull\text{-}out}$). The MEMS devices are then coupled through mutual negative feedback ($w_{12} < 0$, $w_{21} < 0$) and each device is actuated with a square signal with an amplitude equal to its corresponding pull-in voltage, as shown in Fig.4.15(b). Initially, both MEMS devices are actuated with $V = V_{pull\text{-}in}$. However, only $MEMS_1$ is allowed to switch to a high state (ON) while $MEMS_2$ remains in a low state (OFF) because of the negative feedback signal inhibiting the response of $MEMS_2$. Therefore, $MEMS_2$ is only allowed to switch ON when: (1) $MEMS_1$ switches OFF and (2) $MEMS_2$ encounters a high input signal. Both conditions were met at t = 5.5s, causing $MEMS_2$ to switch ON. Similarly, $MEMS_1$ was also inhibited at t = 6.7s, when MEMS2 was active. This behavior demonstrates the production of selective switching (priority bias) due to coupling, where the system only responds to the faster of two signals. This also represents a winner-takes-all (WTA) network, where the activation of one "neuron" leads to the inhibition of other "neurons."

Self-oscillation allows a network of coupled MEMS devices to oscillate using a single DC source, as shown in Fig.4.15(c). This behavior is attainable using $V_1 = V_{pull\text{-}in1}$, $V_2 = 0$, $w_{12} < 0$ *and* $w_{21} > 0$. The positive feedback, w_{21}, ensures that $MEMS_2$ is excited when $MEMS_1$ pulls-in. In constrast, if $MEMS_2$ pulls-in, it inhibits $MEMS_1$ because of its negative feedback, w_{12}. Consequently, this pulls-out $MEMS_1$ and cuts the voltage from $MEMS_2$. Finally, as $MEMS_2$ pulls-out, this allows $MEMS_1$ to pull-in again, thus completing the cycle. The interaction of the two MEMS devices here creates a Hopf bifurcation, which explains the self-oscillation.

Selective switching and self-oscillation were previously reported as characteristics of dynamical neural fields and continuous-time recurrent neural networks (CTRNNs). Thus, a small network of MEMS devices is shown to exhibit some computationally favorable characteristics.

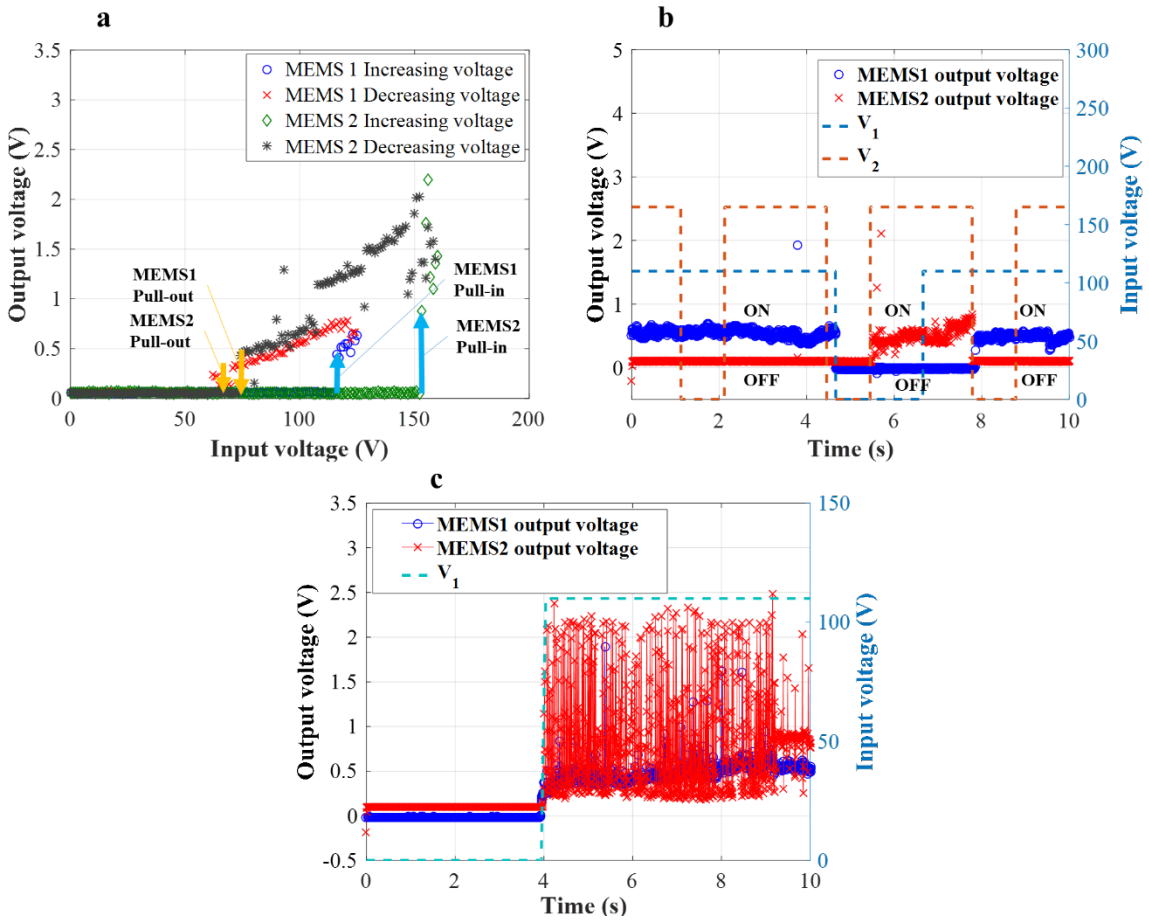


Figure 4.15 The experimental dynamics of two coupled MEMS devices. The MEMS devices are inherently non-oscillatory due to high viscous damping when operated at atmospheric pressure. (a) Device characterization for $MEMS_1$ and $MEMS_2$, showing pull-in (110V and 165V, respectively) and pull-out (66V and 73V, respectively) voltages and regions of hysteresis. (b) The dynamic that arises from coupling the two MEMS through mutual negative feedback ($w_{12} = w_{21} = -1$) showing that if one MEMS device is activated, the other device is attenuated. Thus, the response of the system depends on the timing of the input. $MEMS_1$ is pulled-in initially so it turns ON first. (c) Oscillatory dynamics generated by using positive and negative feedback in the system ($w_{12} = -1, w_{21} = 1$), which leads to limit cycle under DC input.

4.3. Chapter Conclusions

This chapter shows the use of MEMS devices to simulate the behavior of continuous-time recurrent neurons and neuronal populations from the DFT, capturing the behaviors of detection, memory and selection. An analogy between the nonlinear dynamics of CTRN and the nonlinear dynamics of overdamped bistable MEMS structures was made. We have introduced bistability in a MEMS device by triggering its electrical circuit resonance. Operating the MEMS around its circuit's electrical resonance was also shown experimentally to significantly amplify the voltage across the

MEMS. Moreover, we have developed a coupled model for the MEMS circuit accounting for the parasitic components in the circuit and the MEMS equations of motion to simulate a CTRN. Finally, a single CTRN and multiple coupled CTRN behaviors were simulated using a MEMS arched beam.

This chapter shows that a MEMS with electrical resonance activation or initial curvature retains a similar dynamic to that of a CTRN. Thus, making it a prominent candidate as an analog-based building block of a new type of computing unit that is based on the human neuron. Thus, the new MEMS neuron-computing unit could create truly analog brains compounded from devices that respond to stimuli in a similar fashion to human neurons. The use of this new analog computing unit by utilizing its underlying physics can provide a platform to cope with high computing power requirements.

This chapter mainly focuses on recreating the behaviors of individual CTRNs or small CTRNNs using MEMS devices. Chapter 5 builds on the results of this chapter by investigating the computational ability of MEMS CTRNNs to perform both pure computation and colocalized sensing and computing.

Chapter 5

Computation using MEMS networks [APL paper + ASME 2020]

In the context of neural networks, computation mainly involves regression problems and classification problems. In the former, the neural network is expected to find a numerical value, such as predicting future stock prices or expecting the price of houses. The latter problems involve categorizing an event into two or more classes. In robotics, computation may also include trajectory planning and even some aspects of control. In this chapter, a MEMS network is used as an analog CTRNN based on the analogy between individual MEMS and CTRNs that was found in Chapter 4. Two computational tasks are considered here: an active categorical preceptor problem is investigated to test the pure computational ability of MEMS devices; and a simple acceleration waveform classification task is chosen to test the colocalized computational abilities of MEMS sensor CTRNNs.

5.1. Pure Computation Using a MEMS CTRNN

In this section, an active categorical preceptor, proposed by Beer [86] [87], is used to demonstrate object classification and tracking in a MEMS network. The goal of the network is to control a virtual robot (an agent) to catch falling circular objects and avoid falling line objects. This task is illustrated in Fig.5.1,a. The robot is equipped with seven equiangular linear-proximity sensors and two motors, moving the robot transversely. The computational MEMS network is composed of three layers: an input layer (7 MEMS), a computational layer (5 MEMS) and an output layer (2 MEMS), as shown in Fig.5.1,b [86] [87].

For this simulated network, MEMS arches are used due to their bistability via snap-through instability (buckling through the undeformed section). Moreover, MEMS arches maintain their stability beyond snap-through, thus allowing for further deflection, if necessary. The dynamics of a MEMS arch are governed by [74] (4.3).

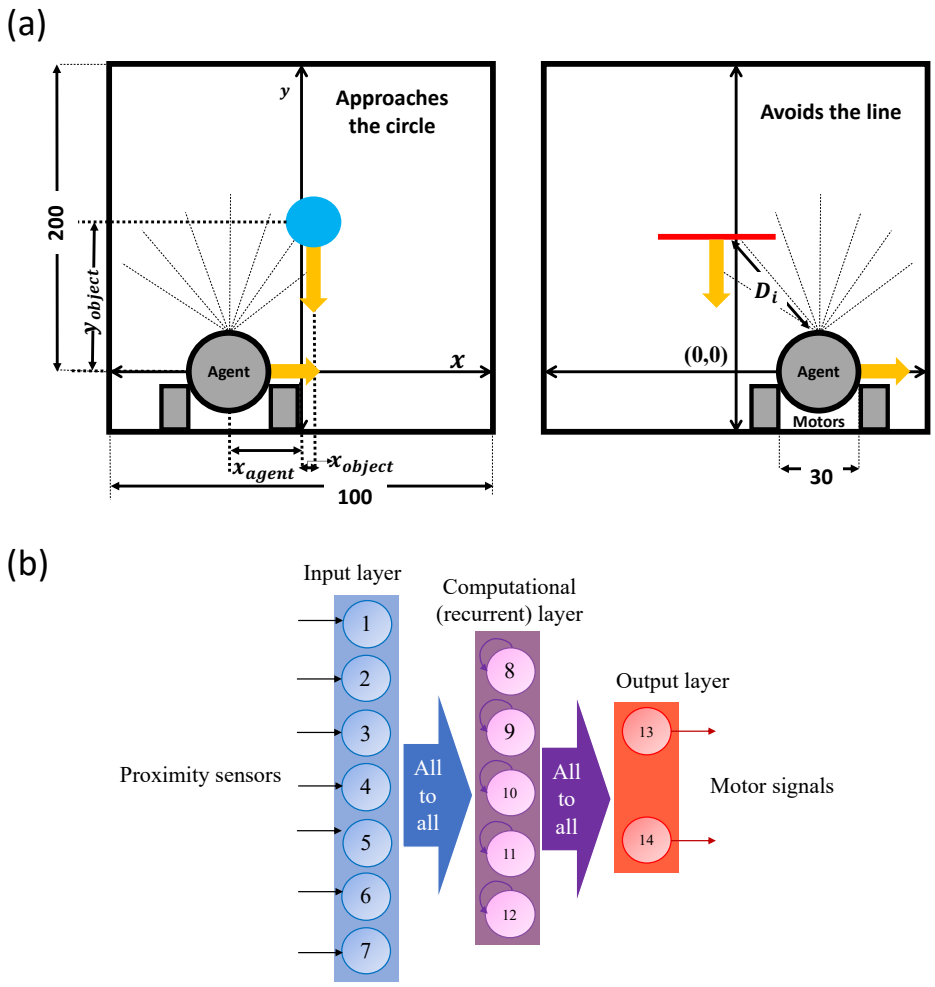


Figure 5.1 (a) Active categorical perception problem. The agent (controlled unit), modeled as a circular object, is equipped with 7 proximity sensors, depicted as dashed lines, is expected to categorize a falling object and act according to its shape. The agent is actuated via two motors attached to each side. (b) The recurrent neural network map used in this study, showing 14 total MEMS neurons: 7 input neurons, 5 computational, recurrent neurons and 2 output neurons. The connection map of the sensors network shows: (1) the one-to-one connection between the proximity sensors' output and the neurons in the input layer as well as the all-to-all connections between the neurons in the input layer. (2) The forward connection between all input neurons to all computational neurons, the all-to-all connections between the computational neurons as well as the recurrent, self-feedback connection in each of the computational neurons. (3) The forward connection between all computational neurons to all output neurons and the all-to-all connection between the output neurons. Finally, the figure shows the output signal of the network to actuate the motor.

The MEMS microbeam dimensions are given in Table 4.1. The initial midpoint elevation b_0 is zero for MEMS in the input and output layer, while it is unspecified in the recurrent layer as it will be optimized in the training process.

The MEMS device is actuated electrostatically with a voltage $V_{MEMS,i}$ causing the i^{th} MEMS to deflect by x_i, measured at the microbeam's midpoint (positive away from the fixed substrate). The voltage across each MEMS device in the network is given by (5.1)

$$V_{MEMS,i} = \begin{cases} V_{0,i} + \sum_{j=1}^{M}\left[w_{sensor,j}V_{sensor,j}(D_j)\delta_{ij}\right] + \sum_{j\neq i}^{n}\left[w_{ij}V_{out,j}(x_j)\right] \ , if\ positive \\ 0, otherwise \end{cases} \quad (5.1)$$

Where $V_{MEMS,i}$ is the effective voltage across the i[th] MEMS, $n = M + R + O$ is the total number of MEMS in the network, where M is the number of input MEMS, R is the number of computational MEMS, and O is the number of output MEMS (in our case $M = 7$, $R = 5$, $O = 2$). $V_{b,i}$ is the bias DC voltage applied to MEMS$_i$, δ_{ij} is the Kronecker delta function, $w_{sensor,j}$ and $V_{sensor,j}(D_j)$ are the weight applied to the input voltage from the j^{th} sensor and the sensor's output voltage as a function of its distance to the object, respectively. w_{ij} is the connection weight between the i^{th} and j^{th} MEMS and $V_{out,j}(x_j) = V_{MEMS,j}\ (x_j - b_0)/(d - b_0)$ is the voltage output of the j^{th} MEMS device as a function of its deflection.

Within the input layer, each MEMS device is connected to a single proximity sensor. The proximity sensors transform the measured distances into a voltage signals following (5.2):

$$V_{sensor,i} = \alpha_{in,i}(D_i + \beta_{in,i}) \quad \text{for} \quad 1 \leq i \leq M \quad (5.2)$$

where $\alpha_{in,i}$ is the i^{th} input sensitivity, associated with the i^{th} MEMS in the network, and $\beta_{in,i}$ the i^{th} bias. The sensor signal is inversely proportional to the distance, producing a signal D_i as given by (5.3):

$$D_i = \alpha_{sensor,i}\left(\frac{|L_{MAX} - L_{ray}|}{\mathrm{L_{MAX}}}\right) \quad (5.3)$$

where α_{sensor} = 10 is the sensor gain, L_{ray} is the distance between the intersection of the linear ray and the falling object and the center point of the agent, and L_{MAX} is defined as the maximum intersection distance, which is set to 220 units in this work.

Similarly, the output of the MEMS network is linearly transformed from a voltage signal to the robot velocity, according to (5.4):

$$\dot{x}_{agent} = \alpha_{vel}(V_{MEMS13} - V_{MEMS14}) \tag{5.4}$$

where x_{agent} is the position of the robotic agent along the x-axis, defined in Fig.5.1,a, the dot operator indicates temporal derivatives, and $\alpha_{vel} = 5$ is the velocity gain. The velocity gain translates the MEMS deflection/capacitance change into agent movements. $V_{MEMS,13}$ and $V_{MEMS,14}$ are the output voltages of the two output-layer MEMS (MEMS13 and MEMS14), respectively, as calculated using (5.5):

$$V_{MEMS,i} = \alpha_{out,i}\left(x_i(t) - \beta_{out,i}\right) \tag{5.5}$$

where $\alpha_{out,i}$ is the output gain of the i^{th} MEMS, $\beta_{out,i}$ is the output bias.

Training the MEMS parameters is achieved by optimizing the input sensitivities ($\alpha_{in,i}$) and biases ($\beta_{in,i}$), coupling weights (w_{ij}), self-feedback through curvature ($b_{o,i}$), voltage biases ($V_{0,i}$) and output biases ($\beta_{out,i}$). Due to the large number of parameters to optimize, a genetic algorithm is chosen as the training scheme using a Python code optimized for parallel processing. All other MEMS parameters are fixed to simplify the training process.

The MEMS network information is stored within genomes, which are vectors of real numbers that contain all of the model parameters. The genomes are trained to maximize the fitness function, *F(x)*, for all test sets. *F(x)* is given by (5.7) using the average fitness function of each set, *f(x),* in (5.6).

$$f_i(x) = \sqrt{\tilde{f}_i^{\,3}}, \text{ where } \tilde{f}_i = \begin{cases} \frac{|x_{obj} - x_{Agent}|}{MAX\ Distance} & object\ is\ a\ line\ in\ scenario\ i \\ 1 - \frac{|x_{obj} - x_{Agent}|}{MAX\ Distance} & object\ is\ \ a\ circle\ in\ scenario\ i \end{cases} \tag{5.6}$$

$$F(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \text{ where } n \text{ is the scenario count in the dataset.} \tag{5.7}$$

The training data set includes the object type as well as x_{agent} and x_{ob}. The fitness function increases when the agent is close to a falling circle or when the agent is far from a falling line at the same vertical position. After evaluating the fitness function of the training data, a group of the best genomes is used to construct the first successful generation. The next generation is created by using a crossover function of the previous generation. This process is iteratively repeated at the end of each training set evaluation until a maximum iteration limit was reached or fitness function reached a satisfactory, predetermined value. The genomes of the final generation with the highest fitness function are stored as the optimized network parameter genomes.

For the studied system, 256 fixed and variable parameters were included in the genome. The variables and their ranges are shown in Table 5.1. The genetic algorithm starts with a population of 3000 initial genomes that were generated randomly within the variable range. The best 100 genomes of each generation are carried out to the new generation by utilizing two-point crossover and a mutation rate of 0.2. The training set is chosen to include 42 different scenarios. In all scenarios, the downward velocity of the object, $\dot{y}_{obj}$, and the agent $\dot{y}_{agent}$ were fixed to 200 and 0 units/s, respectively while the initial starting point of the agent, x_{agent} , and object, x_{obj} , are varied between -50 units and 50 units, and -15 units and 15 units, respectively. Finally, the genetic algorithm problem is solved using a 14 CPU VCore, 32 GB RAM server. The 100-iteration training process, with a concurrency number of 20, took 10 hours to run. In the final generation, the best genome is found to have a fitness function of 0.92.

The simulated results of the trained MEMS network are presented in FIG.5.2 In general, the MEMS network was capable of performing the designated task; the virtual robot minimized the distance to the circular object and maximized the distance to the line object. FIG.5.2,b shows a sample of two successful runs. In both cases, the agent initially moved toward left, then either reversed course to catch the circular object or maintained its motion to maximize its separation with the line object. The object characterization process was performed within one second, as indicated by the minimum

point in the agent's response time when following the circular object in FIG.5.2,c. The figure also shows the significant computational intensity in capturing a circle compared to avoiding a line, as the capturing task requires continuous tracking. In addition, capturing the circle showed greater dynamical richness, including an oscillatory behavior between $MEMS_9$ and $MEMS_{10}$, following the same behavior previously shown in FIG.5.2,c. This oscillatory response propagated to the output MEMS devices ($MEMS_{13}$ and $MEMS_{14}$) and into the motion of the agent.

Table 5.1: Genome variables and their range

Variable	Min Value	Max Value
$\alpha_{in,i}$	1	2.5
$\beta_{in,i}$	-1	4
b_0	0	4.00E-06
$V_{0,i}$	-50	50
w_{ij}	-35	35
$\alpha_{out,i}$	1	15
$\beta_{out,i}$	-1	4

The agent qualitatively behaved as expected in all successful cases. The only case of failure reported here is when the agent attempted to capture a circle when the system's initial conditions are: $x_{agent} = -50$ units and $x_{obj} = 15$ units. In this case, as the agent is at its maximum distance to the left of the object, the initial leftward motion may have caused the agent to lose track of the object, resulting in the erroneous decision. This error may be solved by training the system using test samples with greater separation distances between the agent and object.

5.2. Colocalized Sensing and Computing Using MEMS Sensor Networks

The biggest advantage of using MEMS devices for computing over other analog devices, such as sub-threshold transistors and memristors is their prevalence in smart systems as sensors. Thus, as

MEMS networks demonstrate computational abilities (results of section 5.1) and possess the ability to sense input signals, such as force, acceleration, temperature, humidity, pressure, etc, MEMS sensors may be used as sensors and computing elements simultaneously. Thus, MEMS networks may be used to produce a new generation of smart sensors that output high-level information rather than analog or digital values. For a proof of concept, this section presents a small MEMS accelerometer network that can classify an input acceleration waveform into square waveform or triangle waveform, outputting 0V for the former and 5V for the latter.

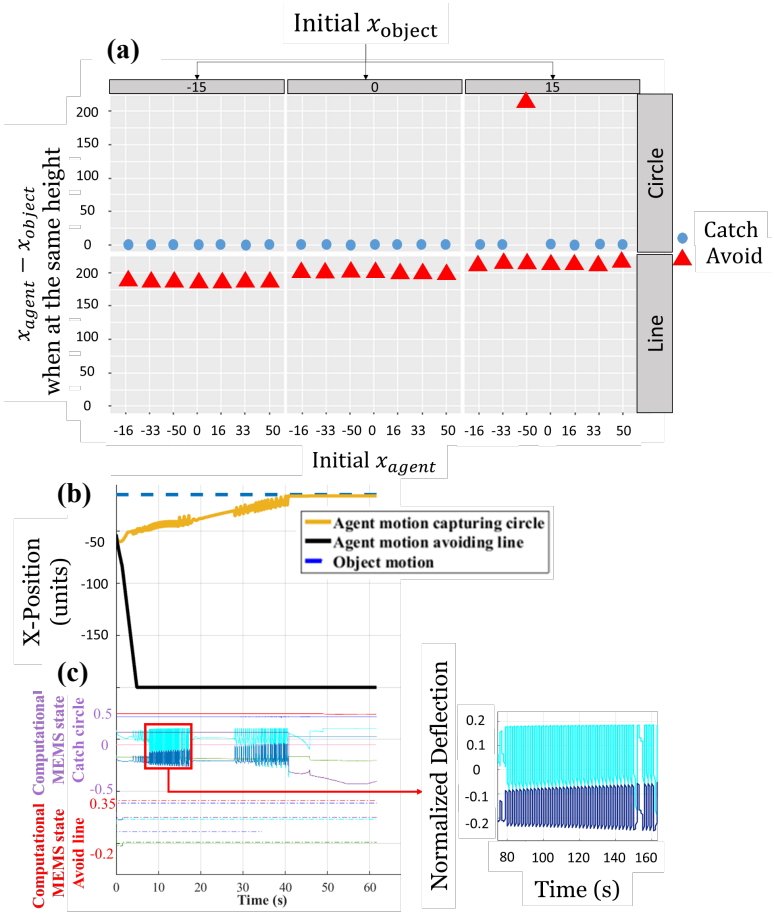


Figure 5.2. Simulated results from the trained MEMS network. (a) Successful and failed attempts to complete the task and the final separation distances between the object and the agent. (b) A sample of the agent's motion when capturing a circle or avoiding a line, with initial conditions $x_{object} = -15$ units, $x_{agent} = -50$ units. Initially, the agent moves to the left, scanning the object, then, according to the object shape, it either reverses to the right to catch the object (circle) or move further away (line). (c) The dynamics of the MEMS in the computational layer normalized to the initial gap (g_0) in the process of catching a falling line object.

Here, for the sake of simplicity, electrostatically driven straight MEMS devices, operated around the pull-in regime, are used as CTRN. The dynamics of each MEMS device within the network of N MEMS accelerometers are given using (5.8)

$$m_{eff,i}\ddot{z}_i(t) + c_{eff,i}\dot{z}_i(t) + k_{eff,i}z_i(t) = \frac{\varepsilon A s_{,i}\left(V_{MEMS,i}(t)\right)^2}{2\left(d_i - z_i(t)\right)^2} - m_{eff,i}\ddot{y}_{base}(t),\ i = 1,2,\dots,N \tag{5.8}$$

This equation is simply a base-excitation-driven SDOF MEMS equation, where $\ddot{y}_{base}(t)$ is the base acceleration. In this equation the absolute position of the MEMS microbeam $x(t)$ is replaced by the relative position of the MEMS microbeam $z(t) = x(t) - y(t)$. To eliminate the pull-in singularity in simulation and avoid electrical contact in practice, stoppers are installed in each MEMS device at a distance $z_{s,i}$. As such, (5.8) is overridden to $z_i(t) = z_{s,i}$ and $\dot{z}_i(t) = 0$ if it was found that $z_i(t) > z_{s,i}$ (section 2.2.3). A schematic of the MEMS device is shown in FIG.5.3.

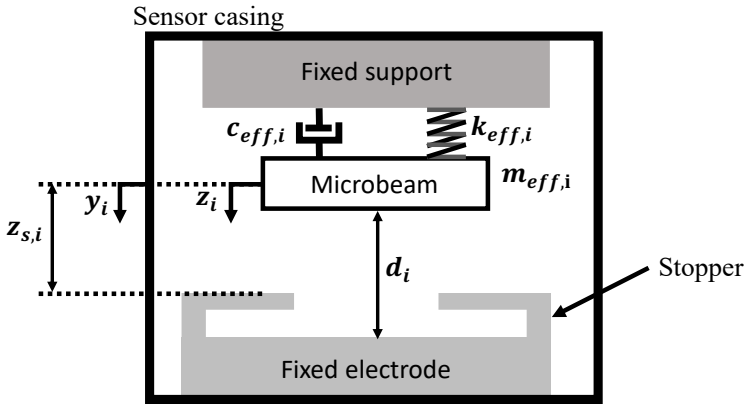


Figure 5.3 The MEMS accelerometers are modelled as single degree-of-freedom spring-mass-damper systems, actuated both via base acceleration and electrostatic attractive forces. Stoppers are utilized in this design to avoid electrical contact.

Coupling MEMS devices is performed electrically using the term $V_{MEMS,i}(t)$:

$$V_{MEMS,i}(t) = V_{0,i} + \sum_{j=1,j\neq i}^{N} w_{ij}V_{out,j}(t) \tag{5.9}$$

where $V_{0,i}$ is the DC bias voltage for MEMS i, w_{ij} is the coupling weight between MEMS i and MEMS j, noting that $w_{ij} \neq w_{ji}$ necessarily, and $V_{out,j}$ is the output voltage of MEMS j given by (5.10):

$$V_{out,j}(t) = V_{0,j}U\big(z_j(t) - z_{s,j}\big) \tag{5.10}$$

where $U(.)$ is a unit step function. Noting that self-connection, typically given by w_{ii}, is essential for computation. While implicit, this recurrent connection is observed in the pull-in regime as evidenced by hysteresis. Here, the MEMS connections are forward and unidirectional (aside from the implicit self-feedback connection). Therefore, $w_{ij} = 0$ if $j > 0$. Moreover, we note that, while the MEMS dynamics are continuous in nature, the state of the MEMS neuron is only interpreted as a binary state in this work due to operation in the pull-in regime. It is still possible to assume that the MEMS state is analog in nature. However, this requires a means of measurement for the response of each MEMS device, defeating the purpose of using MEMS devices as sensors and computing elements simultaneously.

The MEMS network is now tasked with classifying an input waveform into either ‘Square’ signal or ‘Triangle’ signal, as shown in Fig.5.4. The input waveforms are supplied as acceleration waveforms. Here, unlike other physical implementations of neural networks where inputs are electrical signals, the MEMS network used simultaneously performs sensing and computing simultaneously. For the MEMS CTRNN to perform the computational task properly, the size of the network and the connection weights between the MEMS devices are optimized. Optimization was performed manually by starting from a ladder diagram optimization scheme, assuming each MEMS device is a relay. Under that assumption, 5 MEMS devices are required to perform the computational task. The number of MEMS devices required is reduced to 3 by taking advantage of the dynamics of MEMS devices, namely inertia and hysteresis.

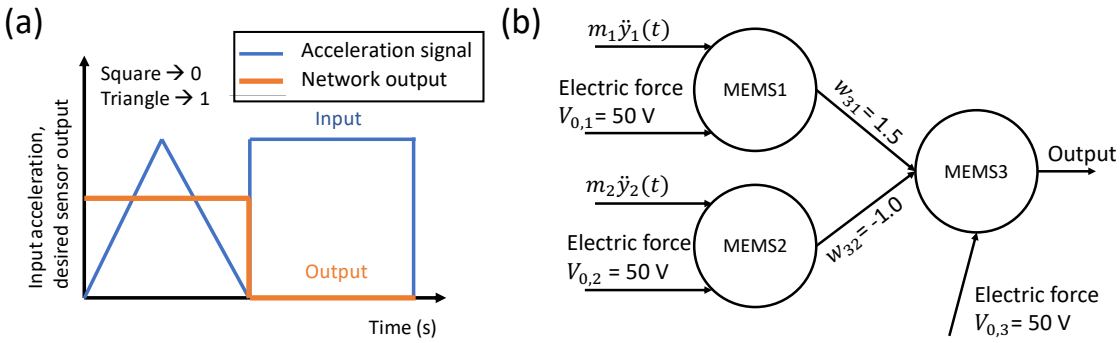


Figure 5.4 Classification task considered in this work. (a) Visualization of the binary classification problem. (b) MEMS network used for classification. The network is composed of three identical devices. Two devices receive an input acceleration signal and one device performs classification.

The bias voltages were chosen such that $V_{0,1} > V_{0,2}$ to force MEMS1 to pull-in ahead of MEMS2 when supplied by a ramped signal. MEMS1 and MEMS2 pull-in nearly simultaneously when a square acceleration signal acts on the CTRNN. The connection weights between the MEMS devices in the network are also optimized manually by taking advantage of the 'selection properties' of a network of a network of CTRNs [32]. Because of selection, the influence of input signals depends on the amplitude of the input signals as well as their temporal order. We note here that, due to our chosen method of weight optimization, the MEMS CTRNN will be able to classify any quasi-static acceleration signal. However, at acceleration frequencies close to the natural frequencies of MEMS1 and MEMS2, this method fails. Other optimization methods would be required to enable classification of such signals.

The constructed network is made of identical MEMS accelerometes. The parameters of the MEMS devices are presented in Table 2.1. Additional information about the sensors used can be found in [73]. The MEMS devices are assumed to be electrically coupled using operational amplifiers to incorporate connection weights. Here, it is assumed that MEMS1 and MEMS2 are input neurons, directly influenced by the acceleration signal. MEMS3, however, is in the computing layer, thus, it is oblivious to the acceleration signal. This can be achieved by rotating MEMS3 such that the acceleration signal is perpendicular to the MEMS motion. This can also be achieved by reducing

the mass of MEMS3 such that the inertial forces are significantly reduced. In this work, the former approach is assumed.

The MEMS CTRNN is subjected to a sequence of a square and triangle signal with an amplitude $\ddot{y}_{base} = -5g$, where g is the gravitational acceleration. The results of the MEMS CTRNN are shown in Fig.5.5. The shock signal excites both MEMS1 and MEMS2 (Fig.5.5,a and Fig.5.5,b, respectively). Initially, when a triangle signal is observed, MEMS1 pulls-in (at around -2g) first due to its higher bias voltage. Consequently, MEMS3 pull-in. When the acceleration signal ramps to -3g, MEMS2 pull-in. Since MEMS2 has a negative connection weight, it reduces $V_{MEMS,3}(t)$. However, this reduction is insufficient to release MEMS3. Thus, MEMS3 remains pulled-in until the acceleration amplitude is reduced to below -2g.

Alternatively, when a square signal is encountered, MEMS1 and MEMS2 experience a sudden and immediate change in amplitude, which results in them pulling-in (nearly) simultaneously. In this case, the voltage acting on MEMS3 is immediately equal to $w_{31}V_{0,1} + w_{3,2}V_{0,2} + V_{0,3}$. By design, this voltage is insufficient to pull-in MEMS3. Therefore, the output of MEMS3 remains low and square classification is performed. Interestingly, MEMS inertia is beneficial in this computing scheme as inertia prevents MEMS3 from pulling-in if MEMS1 pulled in momentarily prior to MEMS2. Moreover, inertia allows this scheme to be performed to classify imperfect square signals, such as signals generated from a shaker which tend to be trapezoidal in shape, assuming the signal ramp is sufficiently steep, since the MEMS devices will slightly lag the input signal.

The results from Fig.5.5 also clearly demonstrate the importance of hysteresis in a MEMS CTRNN as inputs of equal amplitudes may lead to significantly different behaviors depending on past information. (see the areas marked by the red circle and black dashed circle in Fig.5.5,a-d, in which MEMS1 and MEMS2 are simultaneously pulled-in, yet MEMS3 can assume two different configurations).

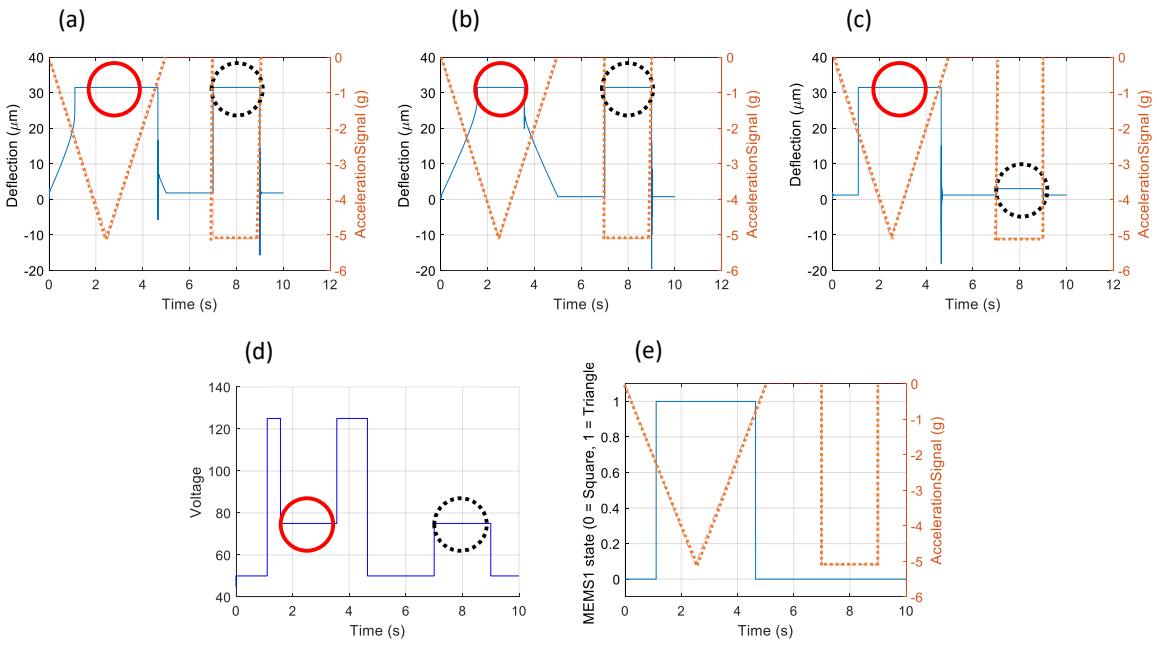


Figure 5.5 Classification test results showing the response of MEMS1 (a), MEMS2 (b) and MEMS3 (c). (d) The effective voltage acting on MEMS3 $V_3(t)$. (e) The state of MEMS3 when subject to a triangle or a square signal. Note: the points marked by red and black circles in (a-d) represent points with similar MEMS1 and MEMS2 states but different MEMS3 states, indicating the importance of memory in a MEMS CTRNN.

5.3. Chapter Conclusions

This chapter demonstrates the computational ability of MEMS sensors networks by performing non-trivial computational tasks in analog, at the sensor level. A network of non-sensory MEMS devices was shown to perform an active categorical perception task with a 92% accuracy, indicating the ability of MEMS networks to perform computation. Moreover, a small network of 3 MEMS accelerometers was capable of performing acceleration waveform classification in analog in the absence of analog-to-digital converters and digital processors.

Computation is performed in the MEMS network by exploiting the inherent nonlinear dynamics of MEMS devices in the pull-in regime to mimic the behavior of a special class of artificial neurons, named continuous-time recurrent neurons (CTRNs).

For simple tasks, training such a binary MEMS network offline is simple using ladder logic as a starting point. Additional modifications by considering MEMS dynamics can reduce the size of the

network. For complicated tasks, such as active categorical perception problems, training, even offline, is computationally expensive. In this chapter, genetic algorithms were used to train the 14-neuron CTRNN. Other training schemes, such as backpropagation through time may be used instead to train the MEMS CTRNN. Aside from offline training, online training may be possible by using memristive devices as capacitors to couple the MEMS devices.

This chapter represents a simple application of intelligent sensory arrays that go beyond simple analog and digital sensing into the domain of classification. Such sensory arrays are expected to significantly reduce the computational load on processors in two ways: perform some computational tasks internally, and allow processors to sleep until a high-level signal of interest triggers an event (such as detecting a triangle signal, rather than relying on a simple signal threshold to trigger the event).

CHAPTER 6

MEMS RESERVOIR COMPUTING FOR SENSING AND COMPUTING

MEMS devices can perform computing by emulating the response of CTRNNs. In this case, MEMS devices are connected in a layered architecture, in which MEMS devices are only allowed to connect to other devices in their immediately succeeding layer. Recurrent connections are inherent in the pull-in, snapthrough and electrical resonance regimes (chapter 4). A critical limitation of MEMS CTRNNs, and of CTRNNs in general, is the difficulty of training. Especially when the number of neurons increases or when the network grows deeper with the introduction of a large number of hidden layers. Additionally, large MEMS CTRNNs requires using a large number of MEMS devices, which may conflict with the size constraints in systems such as wearable electronics or micro-robotics. Thus, MEMS CTRNNs may be appropriate to use for simple applications requiring a moderate number of neurons, or when training is performed offline.

The challenges of MEMS CTRNNs can be addressed by changing the architecture of the MEMS network, creating a ‘reservoir’. By connecting a large network of MEMS devices using random connection weights, the input signals are projected into a higher dimensional space, simplifying the computing process. Indeed, the computing output of this network can be attained using a weighted summation of states of the neurons within the reservoir. In this case, training is performed using linear regression. This architecture is known as Reservoir computing (RC).

Reservoir computing can be further leveraged to reduce the physical size of the MEMS sensing-and-computing unit by using virtual reservoirs, rather than physical reservoirs, allowing one MEMS device to perform the task of hundreds of neurons [39].

This chapter shows the use of a single MEMS device to both computing tasks and colocalized sensing and computing tasks.

6.1. Reservoir Computing using Single MEMS Device

In this section, the pure computational ability of MEMS RC is assessed. Therefore, the MEMS device is used only as a computational unit. The MEMS RC input(s) are electrical signals generated from external sensors(s). To this end, a single degree of freedom MEMS device is used in this section to create the virtual reservoir. The virtual reservoir generates N virtual nodes in a serial fashion, creating one virtual node in θ time steps. The generation process concludes at time $\tau = N\theta$, after which, the MEMS reservoir generates the next time step of the virtual neurons. The nonlinear dynamics of the MEMS device are used to enable high dimensional mapping while the properties of the MEMS device as a dynamical system are used to create the virtual nodes. As explained in section 3.4, maintaining transience is the key to generating the virtual nodes and maintaining memory. To this end, the input signal passes through a modulation circuit (FIG.6.1,a) ahead of being fed to the MEMS device.

The input signal $u(t)$ is sampled and held with a period τ to generate the signal $I(t)$ to reduce the need to read sensor data. The modulated signal is then generated by using a masking signal $m(t)$, such that $J(t) = m(t)I(t)$. The exact generation scheme is explained in section 3.4 and a plot of the input transformation is shown in FIG.6.1,b.

The modulated signal $J(t)$ is then fed to the MEMS device as an electrical signal. Additional input signals are also supplied to the MEMS reservoir to modify its dynamics appropriately. For this dissertation, the voltage supplied to the MEMS reservoir V_{MEMS}, assuming a delayed feedback signal with a gain α and period τ is added to the electrical input, is given by (6.1), if $J(t)$ is added to the DC signal, and given by (6.2) if $J(t)$ is added to the AC signal:

$$V_{MEMS} = \big(V_{DC} + J(t)\big) + V_{AC}\cos(\Omega t) + \alpha x(t-\tau) \tag{6.1}$$

$$V_{MEMS} = V_{DC} + \big(V_{AC} + J(t)\big)\cos(\Omega t) + \alpha x(t-\tau) \tag{6.2}$$

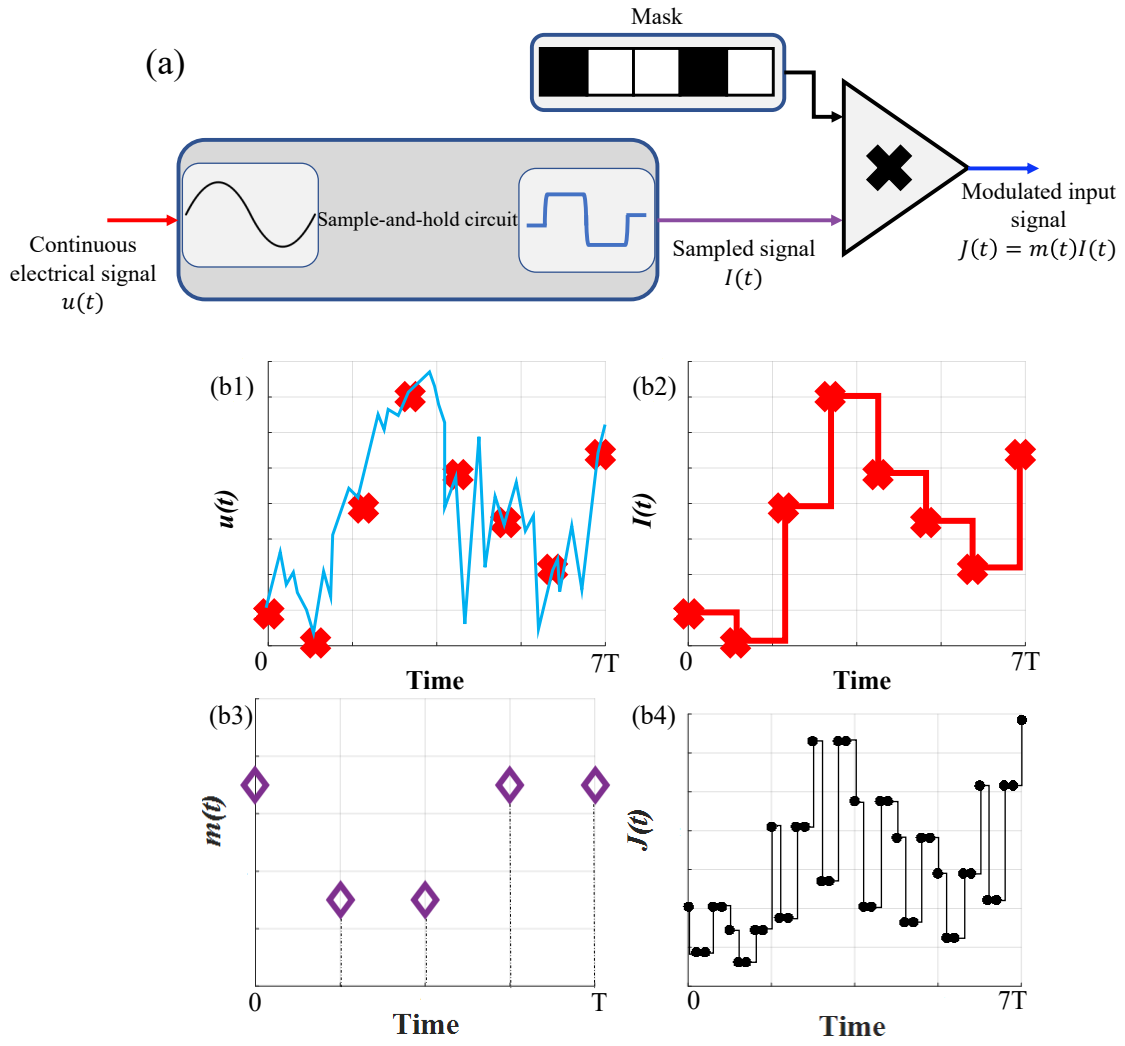


Figure 6.1 Input stage of the reservoir computing setup. (a) Schematics for the modulation circuit. (b) Signal transformations in the modulation circuit.

Delayed feedback is also introduced in the MEMS reservoir circuit to facilitate recurrent connections for each virtual neuron in the reservoir. As the reservoir generates N nodes in τ time units, the delayed feedback is set to τ to ensure self-coupling. The dynamics of the MEMS reservoir vary significantly based on the choice of delay time, as shown in FIG.6.2. In this figure, the delayed feedback loop has a gain of $\alpha = 0.1\, V/\mu m$. Positive delayed feedback is used here as the aim of the loop is to increase the dynamical complexity of the MEMS device rather than stabilizing its orbit. The bifurcation diagram shows the simulated response of a straight SDOF MEMS device with dimensions given in Table 4.1 is simulated. This MEMS device is driven using $V_{AC} = 30\, V$,

$V_{DC} = 58\,V, \Omega = 0.78947\omega_n$ and various delay time τ values. At high τ values, the MEMS device experiences simple sinusoidal oscillation, as indicated by the upper and lower oscillation branches. As τ decreases, the MEMS response goes through a series of period-doubling bifurcations, starting at $\tau = 0.576/\omega_n$, and leading to pull-in.

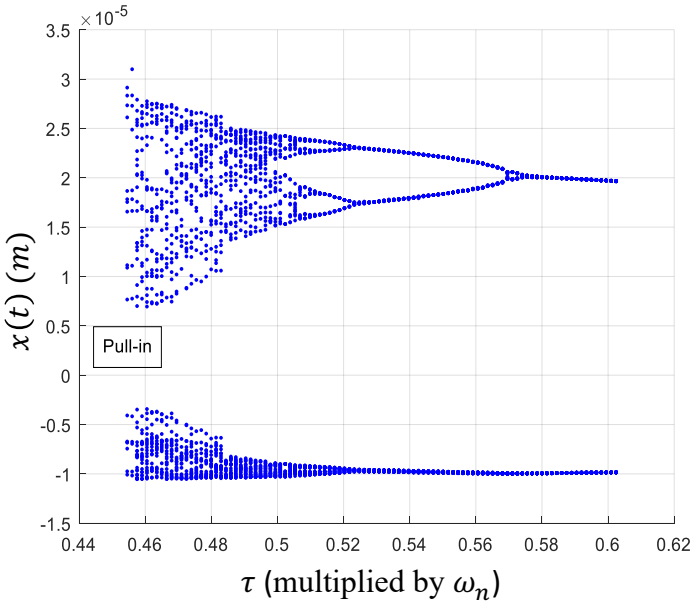


Figure 6.2 Bifurcation diagram of a SDOF straight MEMS device as a function of delay time. Here, $V_{DC} = 30, V_{AC} = 58, \Omega = 0.78947\omega_n$. *The MEMS device is disconnected from the modulated signal in this bifurcation diagram.*

The relationship between τ and θ thus may constraint the reservoir design, as certain operational regimes may only be accessible for low values of τ. However, choosing a small τ along with a large number of virtual neurons N will result in a very small θ. Consequently, as $J(t)$ is typically a piecewise continuous function with a step size of θ, the MEMS device may be excited with very fast signals, to which it will fail to react. This may lead to information loss at the RC level. Therefore, careful analysis of the MEMS reservoir frequency response, the input signal frequency components, the complexity of the computational task and the dynamics of the MEMS device must be performed to choose the best values for τ and θ. The number of virtual neurons is then automatically computed based on τ and θ.

The generation of the neuronal state matrix X is achieved by sampling the MEMS response at θ. For the duration of τ, each sampled $x(t)$ is considered the state of a virtual neuron at the i^{th} time step. The state of each virtual neuron is updated once all virtual neurons have been generated, i.e., after τ time units. Thus, virtual neurons are generated serially in this scheme. This process is visualized in FIG.6.3.

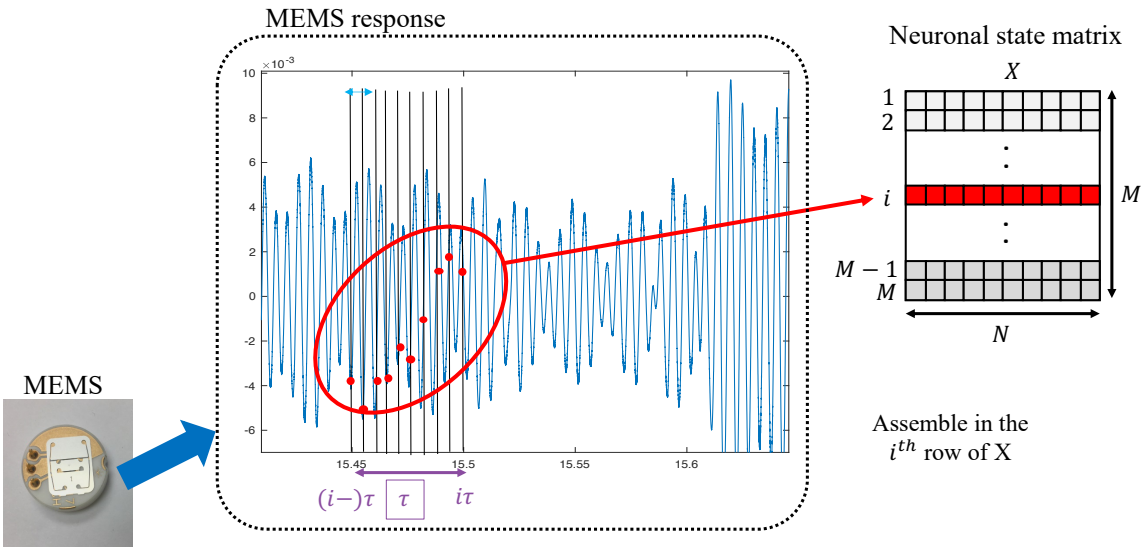


Figure 6.3 The process of constructing the neuronal state matrix from the MEMS response.

Similarly, the output of the RC is only generated once all the virtual neurons have been generated at each time step, i.e., after τ time steps. This output is simply computed using weighted linear summation, as was previously shown in (3.5).

The overall architecture of the MEMS RC scheme is shown in FIG.6.4, showing the input modulation stage (pre-processing), MEMS reservoir stage (processing), and output generation (post-processing) stage. Table 6.1 summarizes the significance of the values of θ and τ in each stage.

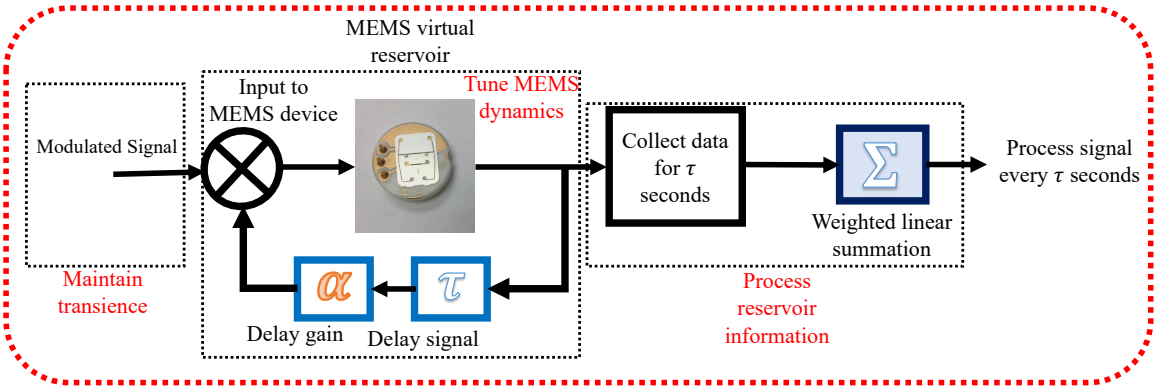


Figure 6.4 A comprehensive schematics of the MEMS RC architecture showing the input modulation stage (pre-processing), MEMS virtual reservoir stage (processing) and weighted summation stage (post-processing).

Table 6.1: Summary of temporal parameter significance in each RC stage

	θ	τ	N
Signal pre-processing	• Separation between masking points.	• Sampling time • Period of the mask	• Number of masking points
Processing	• Virtual node temporal separation	• Processing time • Delay time	• Number of virtual neurons in the reservoir
Post-processing	• MEMS state sampling time	• Computing time	• Number of columns in the neuronal state matrix

The computational abilities of the MEMS RC are tested by performing a simple classification task using the accelerometer device. The classification task involves distinguishing between two input waveforms: rectangular waveform and triangular waveform, which is simple yet non-trivial [35] [88]. To perform this task, the MEMS RC architecture in FIG.6.4 is modified by utilizing two readout circuits rather than one, as shown in FIG.6.5. The readout circuits are used as signal classifiers, such that the rectangle (triangle) classifier outputs +1 if the input waveform is a rectangular (triangular) signal and -1 otherwise.

6.1.1. Classification Using a Simulated MEMS RC

The input signal is supplied as an electric signal to test pure computing. The reservoir parameters are setup as follows: $N = 100, \theta = 1\,ms, \tau = 100\,ms$. The parameters are chosen to ensure transience is maintained while decoupling the virtual nodes from the i timestep from other nodes at the $i + 1$ time step [40]. For tasks with low memory requirement, such as classification, delay-feedback may be forgone with limited impact to RC performance [89]. Hence, α=0 is chosen in this work.

The chosen MEMS device is a commercial accelerometer with the following parameters: $m_{eff} = 106\,mg$, $c_{eff} = 0.78 \times 10^{-3}\,N.s/m$, $k_{eff} = 159.1\,N/m$, $d = 42\,\mu m$, $A_s = 39.6\,mm^2$. While the in-plane dimensions of the MEMS device (surface area) are large, the in-plane electrode separation is sufficiently small to recreate the nonlinear complexities observed in smaller MEMS devices.

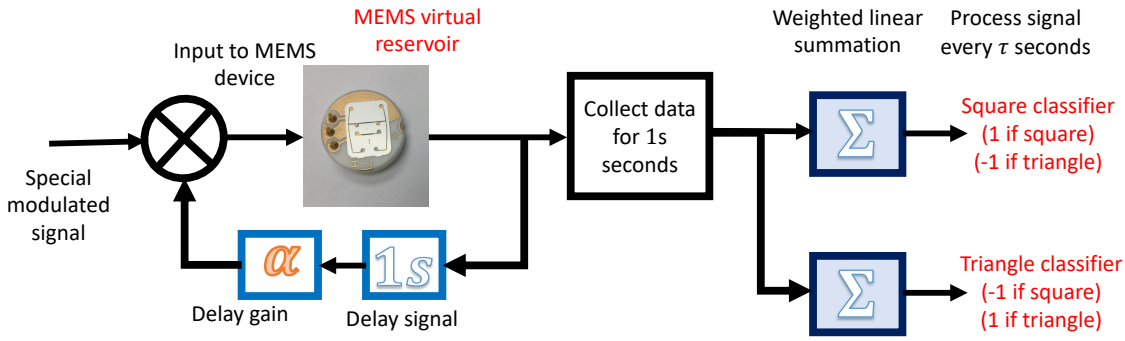


Figure 6.5 RC system design for the classification problem. This binary classification problem requires two readout circuits, one for each class. The classifiers outputs can be computed in parallel using the response of a single MEMS virtual reservoir.

The MEMS device is biased using a biasing voltage of $V_{DC} = 3$ V and $V_{AC} = 0\,V$, amplified using a 20dB amplifier. The modulated input signal is constructed by applying a binary periodic modulation mask on the input waveforms and applied to the biasing voltage signals according to (6.2). The applied modulation mask can assume one of two states: $w_j \in \{0.3,1\}$, which varies every $\theta = 1$ ms, with a 90% chance of taking the higher value of 1. The period of the modulation mask

is $\tau = 100$ ms. The mask was optimized through trial and error. The input electrical waveform is applied prior to the amplifier, representing rectangular and triangular signals with an amplitude of 3 V. The period of each waveform is chosen to be 14τ, which is much slower than the MEMS natural frequency, which prevents the MEMS device from classifying the input signals using a simple frequency response comparison.

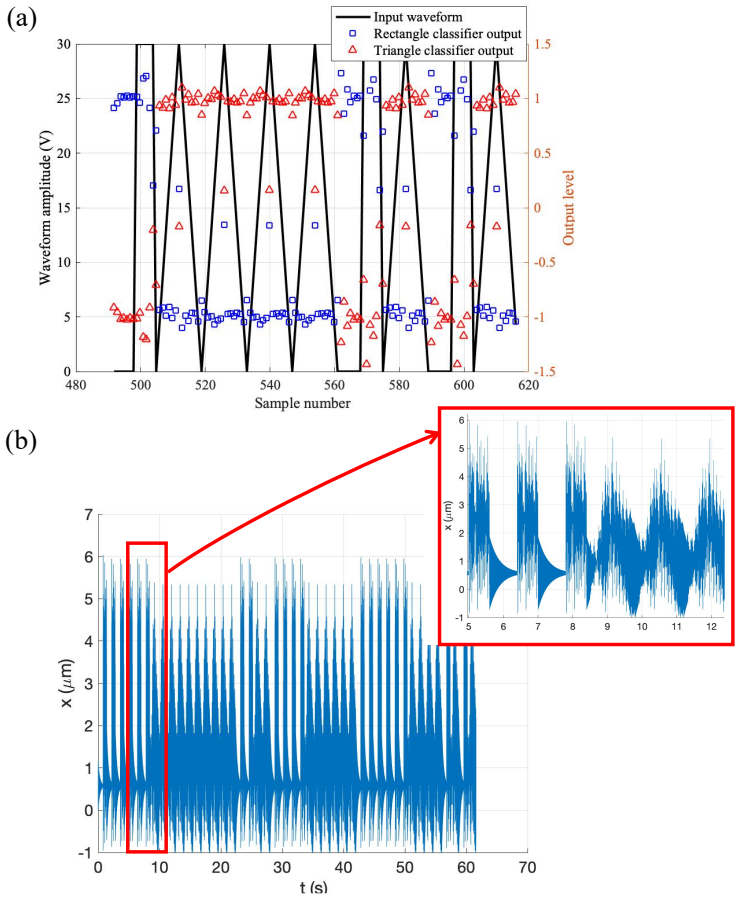


Figure 6.6 Simulated MEMS RC response. (a) Simulated outputs of the rectangle and triangle classifiers, represented by the blue squares and red triangles, respectively. (b) MEMS response due to the modulated signal. Insert: zoomed plot.

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:

$$SuccessRate = \frac{\#TestingSetCorrectClassifications}{\#SamplesInTestingSet} \times 100\% \qquad (6.5)$$

The RC system is subject to a signal train of with a length of 610τ. Around 80% of the MEMS response data points are used to create the training set and 20% of the data points are used for testing. Training is performed using Ridge regression with $k_{reg} = 1 \times 10^{-21}$. The results of this training is shown in FIG.6.6,a where the black solid line represents the input signal, and the blue squares and red triangles represent the outputs of the trained rectangle and triangle classifiers, respectively. In this test, the success rate was found to be 95.4%. Majority of the misclassification occurs at the center of the triangle signal, where the RC classifiers predict rectangular signals, possibly due to the peak amplitude resembling the peak of a square signal after sampling. This misclassification is absent in the triangle signal valley.

The MEMS simulated MEMS response is shown in FIG.6.6,b. It is shown that, despite the input waveform being in the quasi-static MEMS regime, the MEMS device retains a transience due to input modulation.

6.1.1. Experimental RC Classification task

The same task is repeated experimentally using a MEMS commercial accelerometer with the same dimensions. The MEMS device is placed in a vacuum chamber as shown in FIG.6.7 to control the operational pressure. The MEMS device is driven using a data acquisition module, which generates the modulated input signal $J(t)$. This signal is amplified with a gain of 20dB, similar to the simulation. The out-of-plane velocity of the MEMS device is captured using a laser vibrometer, which is later integrated after passing a high-pass filter to find the MEMS deflection. The states of the RC virtual neurons are consequently found by down-sampling the MEMS deflection at a frequency of $1/\theta$. Training is performed offline via Ridge Regression based on the MEMS response to the input signal and the chosen number of nodes, N.

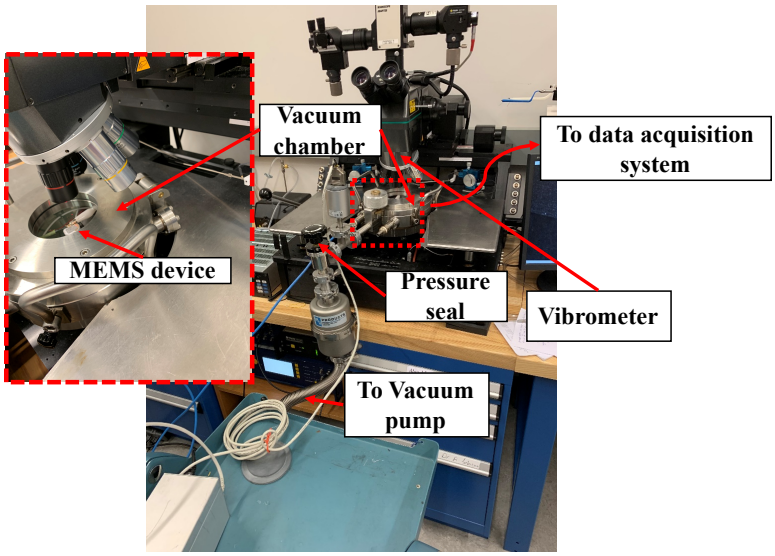


Figure 6.7 Experimental setup for electrical waveform classification. The MEMS device is placed in a vacuum chamber to reduce pressure. The deflection of the MEMS device is attained by integrating the velocity signal from the laser vibrometer.

The MEMS is first tested in the worst-case scenario of operation at atmospheric pressure (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. 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. The virtual states matrix is then split into 80%-20% training-testing sets. The training set is used to optimize the output weights for two classifiers: A rectangle classifier (output = 1 for a rectangle input and -1 for a triangle input); and a triangle classifier (output = 1 for a triangle input and -1 for a rectangle input). Each classifier has a sperate weight matric W_{oR} (rectangle) and W_{oT} (triangle) obtained from Ridge regression. These weights are used to classify the rectangle ($Y_{Rectangle}$) and triangle ($Y_{Triangle}$) output, respectively. The testing set is used to evaluate the success of the RC scheme.

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 (Fig.6.8,a). Low input frequencies produce a quasi-static response, which is unsuitable for the reservoir computing scheme. 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 Fig.6.8,a. A higher average separation distance signifies a more 'confident' classification by the RC. We also test modifying the number of sampled nodes N^* while varying θ to maintain the same τ value. Figure 6.8,b 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. It is noted here that the actual number of virtual neurons in the RC remains constant regardless of N^*, as they are generated by using a mask with values that differ at θ intervals. Thus, down-sampling to produce a lower N^* is a post-processing step to reduce to computational burdens of the system, reducing the size of the X matrix and the need for high-end sampling circuits at the readout portion of the RC.

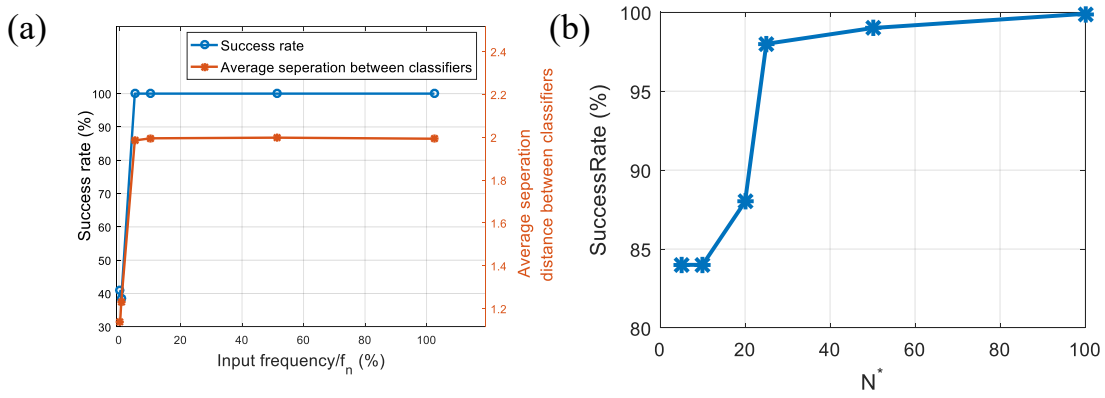


Figure 6.8 (a) Classifier accuracy as a function of the input frequency. At low input frequencies, the classifier fails to classify the input signal. However, as the frequency increases, the reservoir prediction accuracy increases to >99%. (b) MEMS RC classification performance at f/f_n=20% when different N are considered.

The 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 Fig.6.7 and 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 vacuum pump ahead of the experiment. Despite quasi-static input and the introduction of noise, the MEMS RC is capable of performing successful classification. Figure 6.9. shows the classification success rate of the MEMS RC under the influence of pressure variation (99.8%, Fig.6.9,a) and colored noise (99.66%, Fig.6.9,b) at the low input frequency is on-par with previously reported results [35] [88].

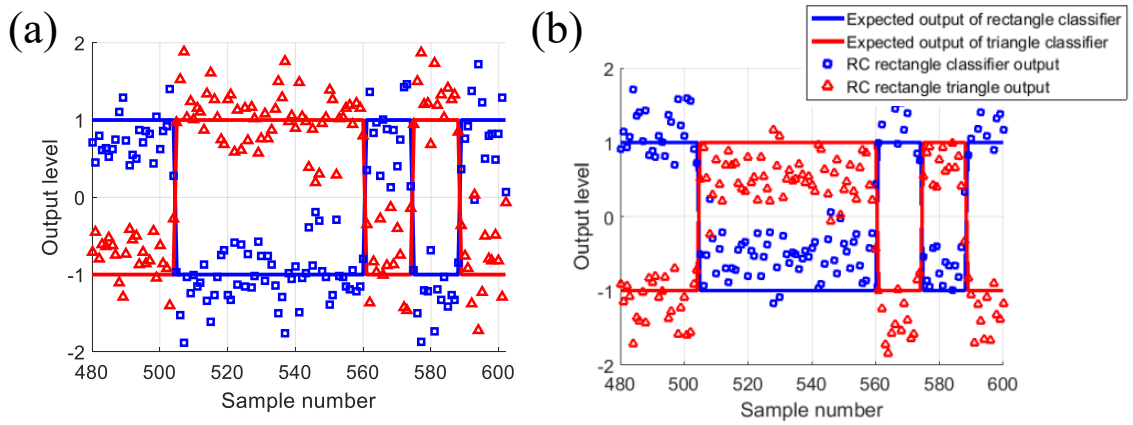


Figure 6.9 Experimental classification of low-frequency signal using the MEMS RC (a) under parameter (pressure) drift, (b) and colored noise. The real-time results of the RC show success rates of 99.8% and 99.66%, respectively.

6.2. Colocalized Sensing-and-Computing using a MEMS Reservoir Computer

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.

However, as environmental signals measured by MEMS sensors are rarely electrical, the current input modulation technique is unsuitable. An alternative approach to input modulation is shown in (6.6), by employing input modulation as a voltage biasing term rather than the traditional input forcing term.

$$J^*(t) = w_j \times V_b,\ (i-1)\tau + (j-1)\theta \le t < (i-1)\tau + j\theta,\ j = 1,2,\dots,N-1 \qquad (6.6)$$

where $J^*(t)$ is the bias time-multiplexing signal, which is piecewise constant for the duration θ, w_j is a period τ mask and V_b is some DC bias applied to the electrostatic MEMS. The time-multiplexed signal is supplied to ensure the MEMS remains in transience to facilitate node coupling, which occurs when $\theta < 1/(2\pi\zeta f_n)$, $\zeta = c_{eff}/(4\pi\, m_{eff}\, f_n)$ is the damping ratio of the system, and $f_n = (1/(2\pi))\left(\sqrt{k_{eff}/m_{eff}}\right)$ is the MEMS fundamental natural frequency. This developed approach of bias time-multiplexing further improves the performance of the colocalized sensing-and-computing RC by eliminating the need for analog-to-digital conversion, which is otherwise necessary for input multiplexing in traditional delay-based RC. It is noted here that $J^*(t)$ is used in-place of $J(t)$ in any consequent colocalized sensing-and-computing test. Furthermore, this approach enables the MEMS device to handle signals that have features with a duration smaller than τ that would otherwise be lost during the sample-and-hold operation.

A simulated response of the MEMS sensor RC is shown in FIG.6.11. Here, the neuronal state matrix N is 620×100. This matrix is split into a training set and a testing set with an 80%-20% ratio. Training is performed using ridge regression. It is noted here that the input signal period is 2000 times slower than τ. Despite that, the MEMS sensor RC manages to perfectly classify the input signal with 100% accuracy.

Next, this simulated task is reproduced experimentally. The MEMS sensor is mounted into a vibration shaker, as shown in FIG.6.11, which can be programmed to generate a square-triangle acceleration waveform using a programmable controller using a built-in adaptive control scheme.

Acquiring the actual MEMS velocity, $\dot{z}(t)$ is performed by measuring the total MEMS velocity $\dot{x}(t)$, using a laser doppler vibrometer, and the shaker velocity $\dot{y}(t)$, by integrating the acceleration measured from an accelerometer mounted on the shaker. The MEMS deflection is then computed as $\dot{z}(t) = \dot{x}(t) - \dot{y}(t)$. The synchronization between $\dot{x}(t)$ and $\dot{y}(t)$ is performed automatically using the shaker controller.

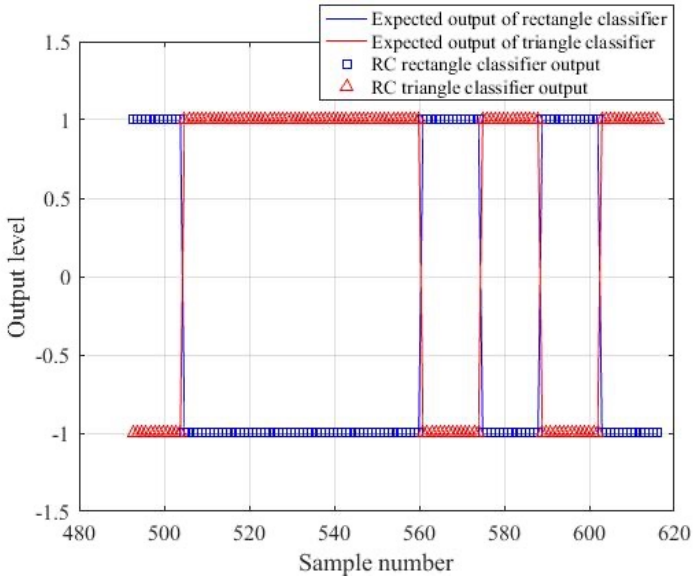


Figure 6.10 MEMS RC output for the colocalized sensing and computing task showing a 100% success in the acceleration waveform classification task.

The sensing and computing task is performed experimentally using an acceleration pulse train with a frequency of $1.1367f_n$ and an amplitude of 5 g, Fig. 6.12(a). The frequency as 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 the modulated signal, which modulates the biasing voltage rather than the input signal. The relative displacement of the MEMS device is shown in Fig. 6.12(b). Figure 6.12(c) 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 [35] [88] and on-par with the results observed from Fig. 6.8. The success of this scheme

shows that the sensing and computing using RC is possible using a slight modification to the input modulation 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 co-local sensing and computing tasks for low-frequency acceleration signals.

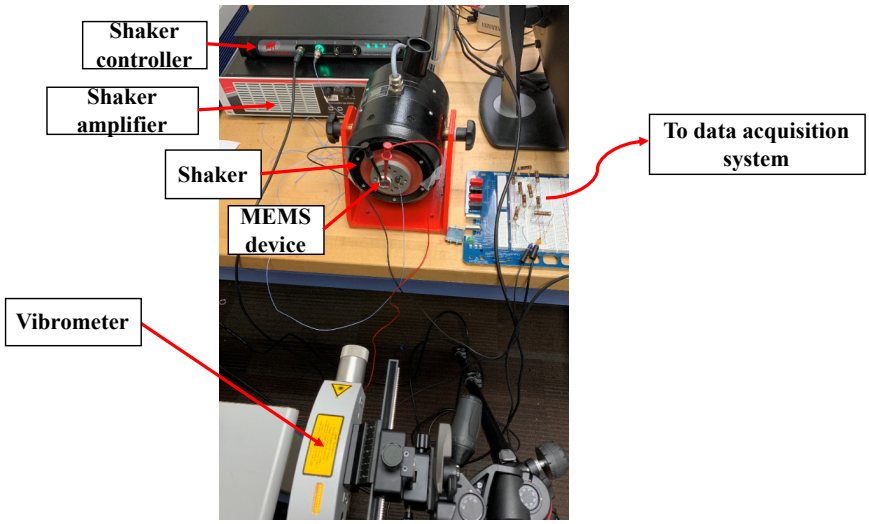


Figure 6.11 Experimental setup for acceleration waveform classification. The MEMS device is fixed on a shaker. The MEMS response is measured as the difference between the microbeam and ceramic base deflections. The shaker is controlled through a dedicated adaptive controller.

6.3. Regression Using Single MEMS Device

Regression is one of the most popular applications of machine learning. The goal of regression is to find predict a numerical value based, rather than predicting a binary class. As there is an infinite number of possible solutions to regression problems, only a single output readout circuit is typically used, rather than a readout circuit for each class, as is used in classification problems.

As a case study for the MEMS reservoir computer, a benchmark regression problem named 'nonlinear auto-regressive moving average' NARMA is studied. In this problem, the MEMS RC is expected to find the response of nonlinear time-series based on current and past inputs [90]. NARMA is a class of problems based on the memory of the time-series. This section focuses on

NARMA10, a dynamical system that is dependent on inputs up to 10-time steps in the past. A NARMA10 dynamical system is modelled using (6.7):

$$y_{k+1} = 0.3y_k + 0.05y_k\left[\sum_{i=0}^{9} y_{k-i}\right] + 1.5u_k u_{k-9} + 0.1 \tag{6.7}$$

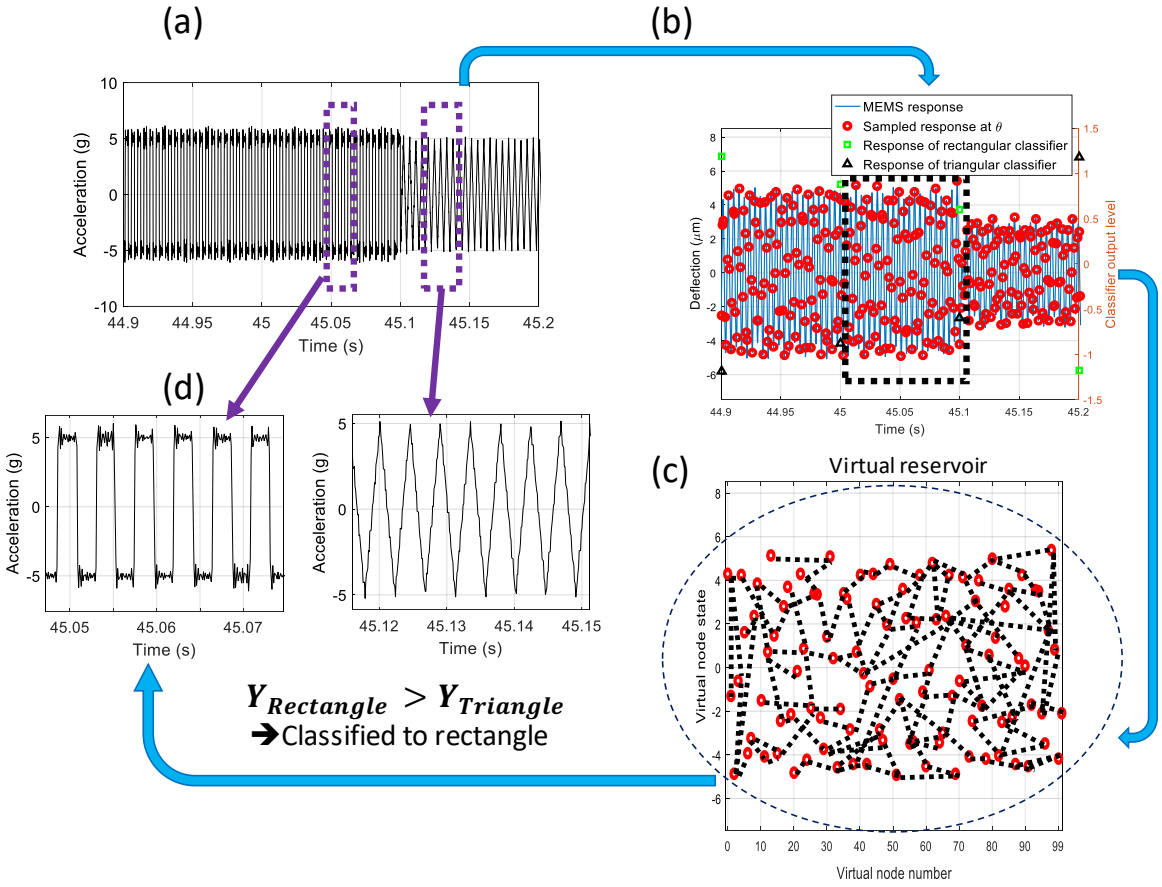


Figure 6.12 MEMS sense-and-compute scheme: (a) A sample of the acceleration signal generated by the shaker. (b) The response of the MEMS device and virtual node extraction. (c) Visualization of the virtual reservoir within the network. The response of the virtual nodes shown in this figure is used to compute the reservoir output after all the virtual nodes are updated. (d) Classification process based on the RC output.

Where y_k is the k^{th} NARMA state and u_k is the input at the k^{th} time step. Following [39], u is chosen to be a random number such that $u_k \in [0,0.2]$. y_k is complicated to fit due to the influence of past values on future responses, which makes this problem a compelling benchmark for nonlinear approximators.

Memory retention is crucial for this application; thus, the MEMS device is operated at a reduced pressure of 20 Pa using V_{DC} = 30 V with no AC voltage. The delay value, τ was chosen to be 0.9s and the feedback gain α = 0.1 V/μm was used. The modulation mask was chosen to be composed of a random sequence of $\pm$0.5 and 0. To ensure sensitivity to inputs, *J(t)* was linearly scaled linearly 5 times. The reservoir in this work was composed of *N = 100* virtual nodes with θ = 0.002 ms.

The NARMA10 simulations were carried out using a sequence of 6000 random inputs (*M= length(u) = 6000*). To ensure good linear fitting, the number of time steps, *M,* must be chosen such that *M* > *N*.

A sequence of 6000 random inputs u to construct the NARMA10 response Y, which represents the target response for the system. The random input is also fed to the MEMS device after modulation to drive the system. The MEMS response to a random input array is shown in Fig.6.13. The response loses its periodicity due to the input modulation and delayed feedback, which is desirable to perform calculations. The response of the MEMS device is sampled at a period of $\theta = 2$ ms and stored in a matrix X. This matrix is split into a training matrix and testing matrix as follows: the first 2000 rows of X are discarded to eliminate the effect of initial conditions, the next 2000 rows of X are chosen as the training set and the final 2000 rows of X are chosen as the testing set. The training set is used to train the Weight Matrix, *W*, through linear regression. Equation 3.4 was used in the training process while setting $k_{reg} = 1\times10^{-21}$.

The performance is evaluated by calculating the normalized root mean square error (NRMSE) as shown in (6.8):

$$NRMSE = \sqrt{\left(\frac{1}{M}\frac{\sum_{i=1}^{M}(s_i - y_i)^2}{(\bar{y})^2}\right)} \tag{6.8}$$

where s_i and y_i are the ith element of the concatenated RC output matrix S and expected output matrix Y, respectively, and $\bar{y}$ is the mean of the vector Y.

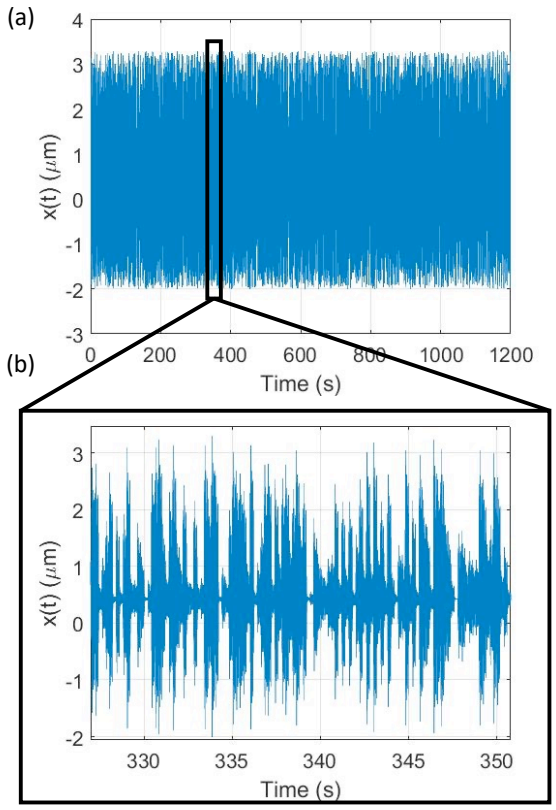


Figure 6.13 (a) sample MEMS response to a random input, u(t). (b) zoomed view of response

Using linear regression to train the weights of the MEMS reservoir using the training set yields NRMSE = 6.18%. The fitting results are shown in Fig.6.14,a by comparing the results of NARMA10 to the results of the MEMS reservoir using the training set again as a test set. Next, the trained weights matrix was tested using the test set (Fig.6.14,b). The result of the test set is NRMSE = 6.43% which is predictably higher than NRMSE from the training set. However, it remains within an acceptable range.

The interaction between nodes occurs due to the delayed feedback used in the reservoir circuit, which also allows past states to visibly influence the MEMS response. Another means of interaction between adjacent nodes occurs automatically through the reliance of each node on the information of previous nodes by virtue of the time-dependence of dynamical systems. However, these interactions are not sufficient to allow the MEMS device to capture the NARMA10 response.

Additional complexity was necessary, thus, unlike the classification task in previous sections, delayed feedback is required.

In the absence of a modulation signal, the MEMS device reach a stable fixed point attractor when actuated, using moderate DC voltage excitation, or a stable periodic orbit if an AC signal is also introduced, after passing through a brief transient state. If the MEMS device is allowed to reach the stable periodic region, the system loses its time dependence, which decouples adjacent modes. To avoid this issue, the separation time between nodes (θ) is chosen such that it is smaller than the characteristic time (time constant) of the MEMS ($\theta < T$), as was used in this section.

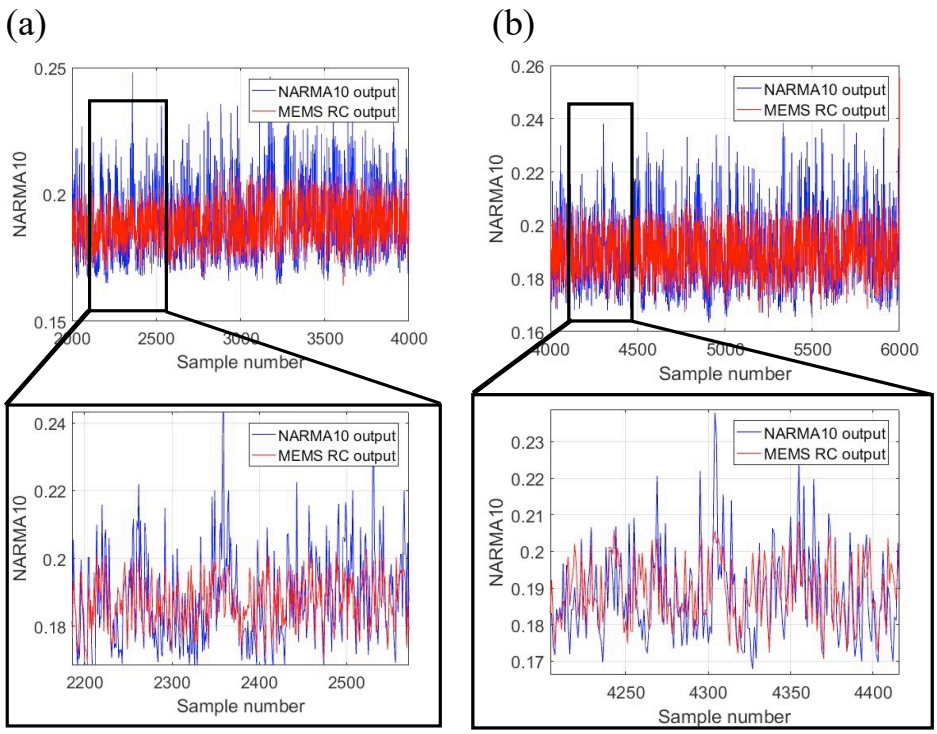


Figure 6.14 NARMA10 approximation (a) training set. (b) Testing set. Inserts: zoomed views.

6.4. Potential of Beam Continuity

MEMS devices have only been modeled as SDOF up to this point. Additional dynamical complexity can be accessed by utilizing the multi-degrees-of-freedom (MDOF) dynamics of continuous MEMS microbeams. If these modeshapes are excited in an appropriate fashion, probing a MEMS microbeam is akin to probing multiple coupled massed connected through elastic (spring) elements. In this case, probing the MEMS device at each time step θ produces more information than what is seen in SDOF MEMS devices or other RC systems. Consequently, it is expected that a lower number of nodes N will be required to yield a satisfactory response for the RC. As computation is performed serially in virtual-reservoir-based RC schemes, this translates to computation faster than SDOF MEMS device. Moreover, since each modeshape has its own modal frequency, the response of the MEMS reservoir will have multiple timescales, which may offer higher computational capabilities (ensembled learners) [45].

Figure 6.15 shows a schematic of a 3 degree-of-freedom MEMS device represented by 3 point masses. The point masses are coupled to each other, as represented by the red springs and dampers. Each degree of freedom correlates to a modal coordinate $u_i(t)$ in the Galerkin discretization. While the modal coordinates are solved for using multiple ordinary differential equations, there exists an internal coupling between the modal coordinates nevertheless.

To demonstrate the potential of using MDOF MEMS devices, we contrast the response of an RC using a MDOF, multi-modeshape continuous MEMS microbeam with the results from an RC using a single-modeshape continuous MEMS microbeam a continuous MEMS microbeam. In both cases, the MEMS microbeam is excited using the input modulation shown in (6.2) to maintain response transience. Moreover, we compare these responses with a continuous MEMS RC device, driven with a modulated signal designed to independently excite multiple modeshapes simultaneously.

In this study, a continuous MEMS arch with dimensions given in Table 4.2 is used. We model the MEMS arch using the first modeshapes to ensure accuracy. As a benchmark, the rectangle-triangle waveform classification task is considered. For this problem, we have found that the parameter

choice: $\theta = 0.2,\ N = 40,\ V_{DC} = 30\,V, V_{AC} = 20\,V,\ f = 24\,, b_0 = 3.47\,\mu m$ are appropriate. Noting that f and θ are non-dimensional parameters, normalized in section 2.2.2. The period of the input signals is fixed here to 7τ, regardless of N to facilitate comparative analysis. A random binary mask with $m_i(t) = \{0,1\}$ is chosen, where the values are equally likely to be chosen. The rectangle and triangle signals have an amplitude of $15\,V$. The amplitude of $J(t)$ is amplified after being calculated using a gain of 15.

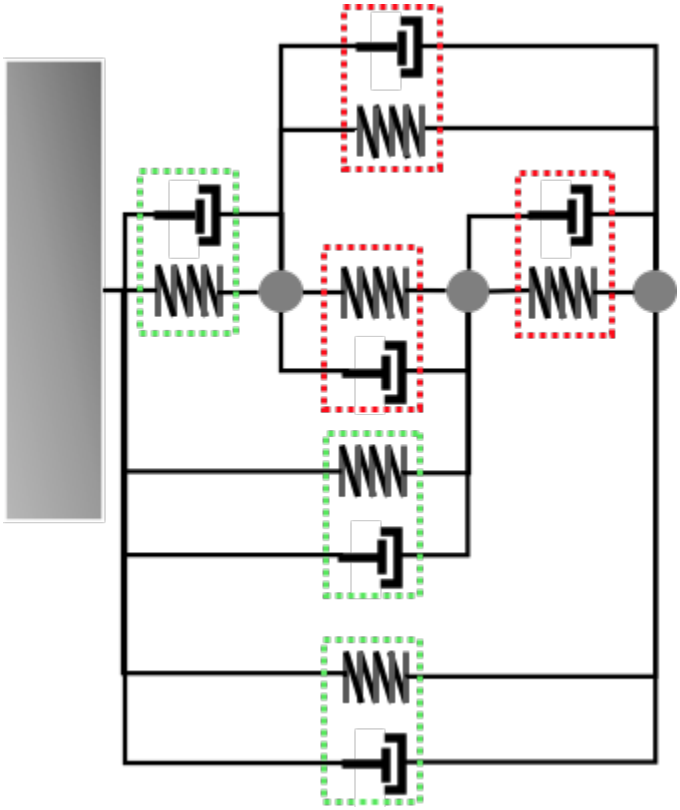


Figure 6.15 Visualization of a continuous MEMS beam composed of three modeshapes. The continuous microbeam can be viewed as a spring-mass-damper system with three point-masses. Interactions between the modeshapes are represented by the springs and dampers enclosed in red dashed boxes, indicating interactions dependent on the modal coordinate $u_i(t)$ and its time derivative $\dot{u}(t)$, respectively. The stiffness and damping coefficient associated with each modeshape are represented by the springs and dampers enclosed in the green dashed box.

Using these parameters, the MEMS response is split into a training set (80%) and a testing set (20%). The MEMS RC is then trained using linear regression to optimized the linear readout circuit weights for both the rectangular classifier and square classifier. Using this approach, when N is chosen to be 40 nodes, a classification success rate of 100% is observed.

To complicate the classification task, the period of input signals is varied across the input waveform. In this case, the normalized period is randomly chosen from {1,2,3,6}, each with equally likely possibility to be chosen. Despite this additional complexity, the MEMS RC accuracy remains 100%, as shown in FIG.6.16.

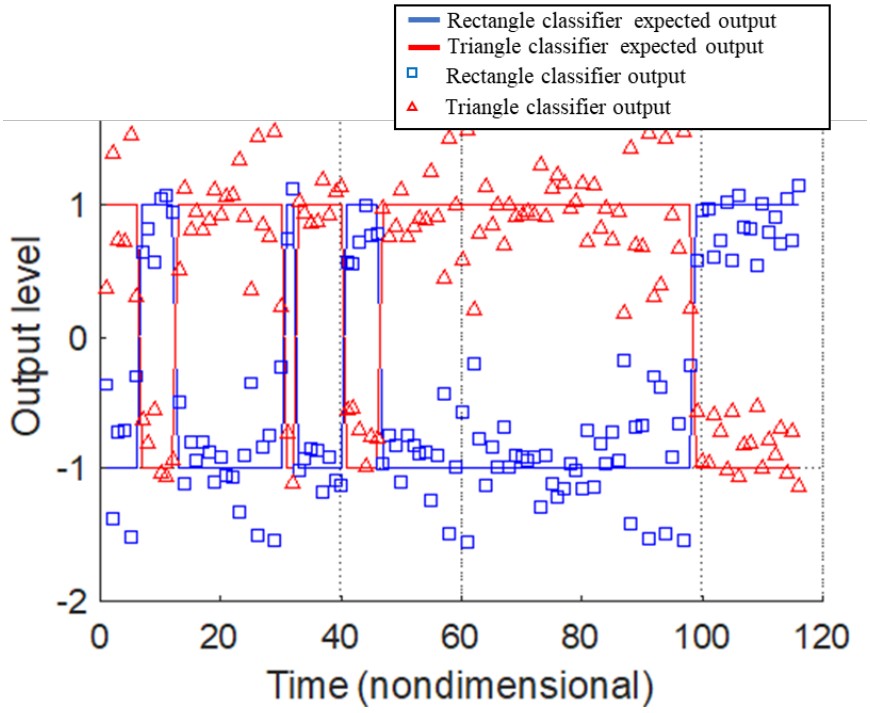


Figure 6.16 Classification results using a continuous microbeam with 5 modeshapes. The input waveform consists of rectangle and triangle signals with different frequencies.

To better assess the performance of the RC, we reduce the number of nodes in the reservoir. Here, we choose a very small number of virtual neurons $N = 1, 5, 10$ and compare the classification accuracy. First, we compare the RC response when using a MEMS beam modeled using 5 modeshapes: (a) when all the AC voltage is supplied at $f = 24$ (near the first modal frequency), (b) when all the AC voltage is supplied at $f = 64$ (near the second modal frequency) and (c) when half the AC amplitude is split in half between $f_1 = 24$ and $f_2 = 64$ (AC components close to both the first and second modeshapes). The results are shown in FIG.6.17.

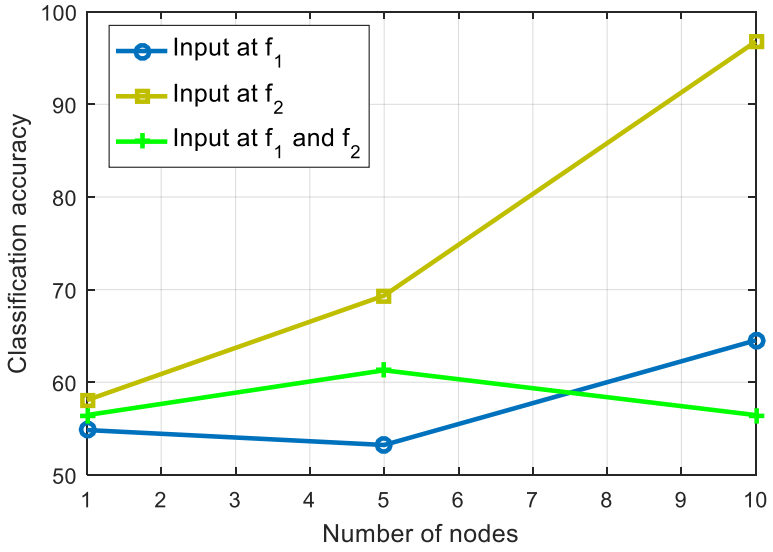


Figure 6.17 Comparison of the classification accuracy of three MEMS RCs, each with a different AC input frequency: $f = f_1 = 24$ (blue), $f = f_2 = 64$ (yellow) and $f = f_1$ and f_2 (green).

The three MEMS RC considered in this study perform slightly better than a coin-flip at $N = 1$. However, as N increases, it appears that the response of the MEMS reservoir with $f = f_2 = 64$ greatly outperforms the other two RCs. Interestingly, at high N, the RC driven using a combination of f_1 and f_2 performed the worst. Each modeshape should be targeted by the modal frequency f_i corresponding to it, in a fashion similar to proof mass producing a high response amplitude at resonance. However, it is noted that this does not truly happen here as electrostatic forcing is quadratic in nature; thus, the frequency components of the input forcing cover a broader spectrum than what would be observed in linear forcing (see chapter 7 for additional information).

The positive effects of modeshape interactions can instead be viewed when observing the MEMS RC performance when $f = f_2$ is chosen. The interactions between the modeshapes is not explicit here. However, implicitly, the modeshapes interact with each other to improve the MEMS response. This may be more obvious when exciting around the second modal frequency compared to the first modal frequency because the first modal frequency is typically dominant in clamped-clamped structures, thus the higher order modeshapes, especially even ones, are overshadowed.

For the sake of ensuring accuracy in the previous sections, we also compare the response of the MEMS RC with 5 modeshapes with that of a MEMS RC with one modeshape to observe the response differences. Figure 6.18 shows that driving the MEMS RC at $f = f_1$ yields worse results than MEMS RC driven at f_2, whether the MEMS device is modeled as a SDOF or a MDOF. The only difference in response between the SDOF MEMS RC and MDOF MEMS RC at f_1 is a constant offset due to the influence of the disregarded modeshapes. The effects of these modeshapes on the RC response appear to be negative, supporting our previous findings.

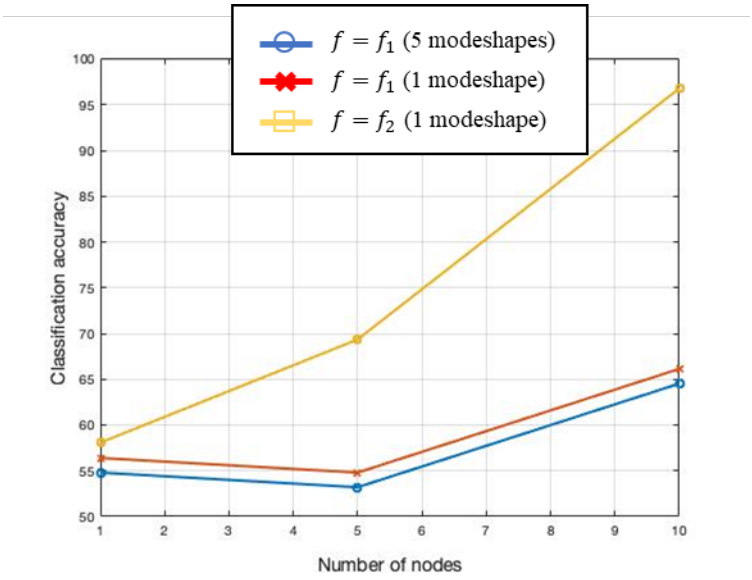


Figure 6.18 Comparison of the classification accuracy of three MEMS RCs: $f = f_1 = 24$ (MDOF, blue), $f = f_1$ (red, SDOF) and $f = f_1$ (MDOF, yellow).

The results of this figure show that our RC analysis in previous sections remains valid even when assuming a SDOF MEMS device. The accuracy discrepancy between simulated and experimental results may be due to disregarding higher order modshapes.

The large number of neurons in RC is needed for high dimensional projection. However, if high dimensional projection is attained through the system itself, there may be less need for a large number of virtual in the RC. Since the processing time is equal to τ in reservoir computing schemes, decreasing N would also reduce the processing time. This can significantly improve the

performance of the MEMS RC. Moreover, as mentioned previously, the use of multiple 'virtual reservoirs' for each modeshape introduces a multi-timescale property to the MEMS RC, which may enable the MEMS to perform intricate tasks by extracting short signatures from the low-frequency modeshapes and long signatures from high-frequency modeshapes in a fashion similar to the time-constant tuning of CTRNNs.

6.4. Chapter Conclusions

The use of a single MEMS device to emulate a large neuronal reservoir is presented in this chapter. The MEMS reservoir was used to perform a classification task and a standard regression benchmark test: approximating the response of a NARMA10 system.

The MEMS device operates as a reservoir of *N* nodes by creating temporally separated virtual nodes. This is achieved using the modulation mask *m(t)*. The interaction between nodes occurs due to the delayed feedback used in the reservoir circuit, which also allows past states to visibly influence the MEMS response. Another means of interaction between adjacent nodes occurs automatically through the reliance of each node on the information of previous nodes by virtue of the time-dependence of dynamical systems. We note here that MEMS devices reach a stable limit cycle when actuated, using moderate AC and DC voltages excitation, after passing through a brief transient state. If the MEMS device is allowed to reach the stable periodic region, the system loses its time dependence, which decouples adjacent modes. To avoid this issue, the separation time between nodes (θ) is chosen such that it is smaller than the characteristic time (time constant) of the MEMS ($\theta < \tau$).

The standard MEMS RC was shown to have a perform in the classification and regression tasks with over 99% classification accuracy for the former and RMSE = 6.43% for the latter. The standard RC scheme has been modified in this section to enable colocalized sensing-and-computing, eliminating the need for sample-and-hold circuit, analog-to-digital converters and external sensors

without sacrificing performance accuracy. This approach may enable the creation of new generations of smart, RC-based sensors, without using large networks of sensory elements. Such sensors will entirely utilize transience, increasing their speeds. Such sensors may be capable of performing edge computation by performing classification and/or prediction. For example, such 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. The results in this chapter show experimentally that the collocal sensing-and-computing through MEMS devices result in good noise resistance, which is shows promise.

Finally, this chapter presents the use of multi-modeshape continuous MEMS devices to enrich the response of the MEMS RC by introducing inherent modeshape coupling within the MEMS RC. This coupling is shown to improve the response of the MEMS RC, resulting in a higher degree of high dimensional mapping, evidenced by the improved RC accuracy even at lower number of virtual neurons. The use of multi-modeshape continuous MEMS RCs also introduces multi-timescale responses associated with each modeshape, which may enable intricate computation, as evidenced from recent research results showing the effectiveness of using multiple reservoirs with different time scales.

Despite these promising findings, the potential of using MEMS devices for colocalized sensing-and-computing is hampered by the need for high voltages to drive MEMS devices, especially in nonlinear regimes, which are needed for RC. To address this critical challenge, chapter 7 introduces a passive means of response amplification in MEMS devices using ‘double resonance drive’. As a byproduct, the next chapter also explains the ability of MEMS devices, driven at electrical resonance to retain bistability, which was shown in chapter 4.

CHAPTER 7

DOUBLE RESONANCE EXCITATION

Electrostatically actuated MEMS sensors and actuators are extremely energy due to the low current flow in capacitive elements. This makes them convenient to use in smart systems, wearable devices and non-conventional robotics (soft-, micro-robotics). Despite that, electrostatic MEMS devices require a significant amount of voltage to drive, in the order of tens to a few hundred volts. Especially when MEMS devices are to operate in the nonlinear regimes.

To amplify the output signal of MEMS resonant devices several approaches have been utilized, including driving them around their mechanical resonance frequency, optimizing designs [91] [92] [93] by increasing the MEMS structure surface area, reducing its stiffness, or narrowing the gap between the stationary and movable electrodes. However, these methods were not effective to boost the device response while reducing their operating voltage to a level compatible with complementary metal-oxide semiconductor (CMOS) technology. Moreover, most of these methods increase squeeze film damping [94], which is extremely detrimental in applications such as reservoir computing, and the risk of electrode stiction [95]. Other research has focused on utilizing parametric nonlinear resonance to increase MEMS dynamic deflection and enhance the output voltage [96] [97]. However, parametric resonance activation requires complex actuation techniques to modulate the stiffness and strict low damping conditions; and hence is limited for specific applications.

Utilizing mechanical resonance is a common way of amplifying the response of MEMS structure. Previous works extended this concept using multi-frequency excitation signals to increase MEMS filter bandwidth [98], the signal to noise ratio in micro-gyroscope applications, [99] and the harvested energy in MEMS harvester [100]. Finally, as MEMS devices also act as capacitors; electrical resonance was utilized for detection through electrical resonant frequency shift in an RLC

circuit [101] and to amplify the MEMS response by forming an LC tank circuit or a resonant drive circuit [102] [103]. Triggering electrical resonance in such a circuit leads to a large voltage amplification across the MEMS device. However, due to the mismatch between the mechanical and electrical resonance frequencies, only static amplification can be achieved using a typical excitation signal.

In this chapter, the MEMS response is amplified simultaneously activating its electrical and mechanical resonances (double resonance actuation). This developed amplification scheme is operational even when frequency mismatch exists by using a multi-frequency signal [104]. Moreover, the response change of MEMS devices due to electrical resonance drive is investigated.

7.1. Double Resonance Excitation Introduction

The dynamics of a single degree of freedom (SDOF) MEMS device are introduced in Chapter 2 by assuming the MEMS device is a perfect capacitor and assuming the lack of any parasitic components in the MEMS circuit. This model is valid for operational frequencies significantly lower than the electrical resonance frequency of the circuit. Thus, researchers often limit the operational range of frequencies to lower than the electrical resonance.

To extend the range of the electrical model, one must consider the parasitic components of the resonator (Inductance – L_s, Resistance – [$R_{dielectric}$, R_{plate} , R_{wires} (very small)] and Capacitance - C_p) in the model as shown in Fig.7.1,a. In the figure, C_{MEMS} is the variable capacitance and is the sensing element of the circuit. We note that the series R_{plate} and R_{wires} are very small and can be neglected in a circuit with external series resistance. Moreover, the parallel $R_{dielectric}$ is very large and can be assumed an open circuit if the applied voltage is smaller than the breakdown voltage of the material. Therefore, we only consider these parasitic components L_s and C_p in this study.

Utilizing the electrical resonance frequency requires building a resonance LC tank circuit. The MEMS circuit evidentially contains a small parasitic inductance. However, for practical use, the

addition of an external inductance $L_{external}$ is necessary to reduce the system uncertainty and reduce the electrical resonance frequency. An external additional resistance $R_{external}$ is also convenient to add to reduce current flow and avoid creating a short-circuit upon pull-in. In such a circuit, the total series inductance L_e and resistance R_e are given by (7.1) and (7.2), respectively

$$L_e = L_s + L_{external} \tag{7.1}$$

$$R_e = R_L + R_{external} \tag{7.2}$$

Where L_s is the series parasitic inductance and R_L is a parasitic capacitance introduced with the addition of the external inductance. The MEMS circuit can be reduced into the simple form in FIG.7.1,b by disregarding some of the small parasitic components.

In this model, the total MEMS capacitance C_{tot} is given by (7.3)

$$C_{tot} = C_p + C_{MEMS} \tag{7.3}$$

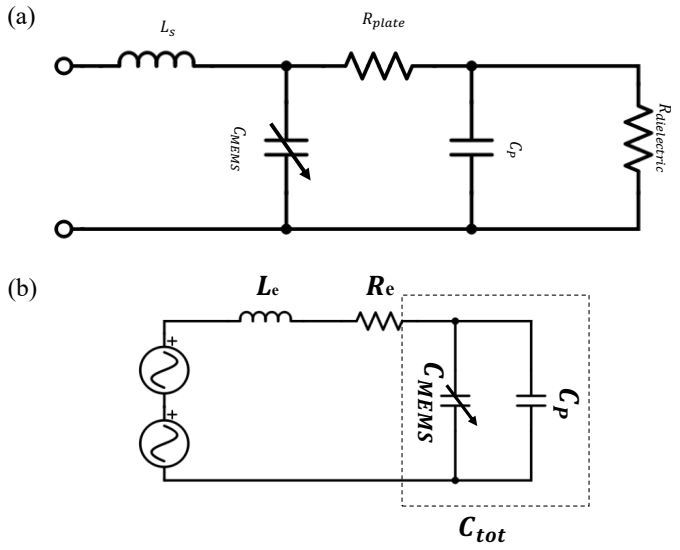


Figure 7.1 (a) A schematics for the equivalent circuit. The MEMS device is modeled as an imperfect capacitance with a small series (lead) inductance (L_s) and a variable capacitance (C_{MEMS}) reflecting the change in capacitance because of the motion, parallel parasitic capacitance (C_p), a negligible plate resistance (R_{plate}) and an almost infinite parallel dielectric resistance ($R_{dielectric}$). All components aside from C_{MEMS} and C_p are negligible in the model. (b) RLC circuit consisting of the MEMS and an external resistance $R_{external}$ and inductance $L_{external}$. In this figure, R_e and L_e are the equivalent resistance and inductance in the MEMS circuit, accounting for the external components.

Where C_{MEMS} is the parallel plate capacitance between the moving electrode and the stationary electrode. If fringing fields are neglected, C_{MEMS} is given by (7.4)

$$C_{MEMS} = \frac{\varepsilon A_s}{d-x(t)} = C_0 \frac{d}{d-x} \tag{7.4}$$

Where C_0 is the nominal capacitance. From (7.3) and (7.4), it is clear that the MEMS capacitance reduces to a variable parallel plate capacitance assuming no parasitic components in the circuit.

The simplified series resonating RLC (resistor-inductor-capacitor) circuit in FIG.7.1,b with an electrical resonance frequency $f_{electrical}$ govern by (7.5):

$$f_{electrial} = \frac{1}{2\pi} \frac{1}{\sqrt{L_e C_{tot}}} \tag{7.5}$$

This consideration results in a series RLC circuit govern by (7.6):

$$L_e \, \ddot{Q}(t) + R_e \, \dot{Q}(t) + \frac{1}{C_{tot}} \, Q(t) = V_{in}(t) \tag{7.6}$$

Where $V_{in}(t)$ is the total input voltage applied to the RLC circuit at time t, $Q(t)$ is the charge stored at the MEMS capacitor electrodes and the dot operator represents temporal derivatives. As the electrical system is governed by a second-order differential equation, if the system is sufficiently underdamped, it exhibits a large response around its electrical resonance frequency. This leads to a large build-up of charge across the capacitances in the circuit. This corresponds to a voltage amplification across the MEMS device, V_{MEMS}. To find V_{MEMS}, (7.6) and (3.1) are solved simultaenously.

Considering the cases where the electrical resonance is significantly far from the mechanical resonance, the following uncoupled, simplified, model can be alternatively used to compute the MEMS effective voltage V_{MEMS}:

By studying the total electrical impedance Z_{eq} of the circuit, we find that:

$$Z_{eq} = R_e + X_e = R_e + \frac{j\left((2\pi f)^2 L_e C_{tot} - 1\right)}{(2\pi f) C_{tot}} = \frac{(2\pi f) R_e C_{tot} + j\left((2\pi f)^2 L_e C_{tot} - 1\right)}{(2\pi f) C_{tot}} \quad (7.7)$$

Where X_e is the equivalent reactance of the circuit, f is the AC frequency and j is the imaginary number. Furthermore, under the assumption the MEMS device mechanical resonance frequency is far lower than the electrical resonance frequency of the RLC circuit, it is possible to solve for V_{MEMS} using voltage division according to (7.8-7.10):

$$V_{MEMS} = V_C = |V_C| \cos(2\pi f t - \phi) \quad (7.8)$$

$$\left|\frac{V_C}{V_{in}}\right| = \frac{1}{\sqrt{\left((2\pi f) R_e C_{tot}\right)^2 + \left((2\pi f)^2 L_e C_{tot} - 1\right)^2}} \quad (7.9)$$

$$\phi = \tan^{-1}\left(\frac{(2\pi f)^2 L_e C_{tot} - 1}{(2\pi f) R_e C_{tot}}\right) \quad (7.10)$$

The maximum voltage computed in (7.9) occurs around the electrical resonance frequency, f_e. The actual frequency corresponding to the amplitude peak depends on the circuit damping, and hence, the total resistance R_e.

By analyzing equations (3.2) and (7.6), the force term of the mechanical system is a quadratic function with respect to the voltage while the force term of the electrical system is a linear function with respect to the voltage. Therefore, it is possible to activate the electrical resonance using a signal with at least one frequency component at the electrical resonance frequency. But to activate the mechanical resonance, an input voltage signal composed of two frequency components is proposed:

$$V = V_{AC1} Cos 2\pi f_1 t + V_{AC2} Cos 2\pi f_2 t \quad (7.11)$$

Then the electrostatic forcing becomes:

$$F_e(V) = \frac{\varepsilon A_s}{2(d-x)^2}\left(\frac{V_{AC1}^2}{2}[1 + \cos(4\pi f_1 t)] + \frac{V_{AC2}^2}{2}[1 + \cos(4\pi f_2 t)]\right.$$

$$+V_{AC1}V_{AC2}[\cos\{2\pi(f_1 - f_2)t\} + \cos\{2\pi(f_1 + f_2)t\}]) \quad (7.12)$$

Therefore, the frequency components of the force term acting on the mechanical system are $2f_1$, $2f_2$, f_1- f_2, f_1+ f_2 as well as the DC component. Any of these frequency components of activates the mechanical resonance when it equals the magnitude of the primary resonance frequency. Thus, using a mixed-frequency signal with one component equal to the electrical resonance frequency while the other frequency is any of the frequency combinations results in at least one frequency equivalent to the mechanical resonance frequency and activates both resonances simultaneously. Here, the tested resonator has a very low resonance frequency that is far smaller than the electrical resonance frequency. Thus, the proper frequency component to be excited is f_1- f_2 which allows both f_1 and f_2 to activate the electrical resonance.

The double resonance actuation scheme is used in this chapter on two MEMS devices: a large commercial accelerometer, which was previously used within this dissertation; and a small MEMS device. The former has a very small mechanical resonance frequency, which causes a large frequency mismatch, while the latter has a relatively large mechanical resonance frequency, which allows a simplified double resonance actuation scheme.

7.2. Double Resonance in Large MEMS Structures

A Sensata™ MEMS accelerometer consisting of an out-of-plane-displacement, electrostatic double-cantilever is used as the experimental design. The parameters of the structure can be found in Table 2.1. Using Lyncee Tec's digital holographic microscope (DHM), the out-of-plane displacement that corresponds to the mechanical response of the resonator under investigation is measured. The experimental setup is shown in Figure 7.2. The stroboscopic module drives the circuit, composed of the resonator and an external inductor. The input signal and the voltage across the resonator are measured through different virtual channels in a data acquisition system.

7.2.1. Experimental Characterization

Initially, the device is driven at atmospheric air and with a low input voltage to reduce the vibration and the transient deflection of the resonator, therefore, reducing the structure to a constant capacitance and measuring the gain of the circuit shown in FIG.7.3. The electrical resonance frequency, corresponding to a gain of *13* times, is found to be *64.6 kHz*. Using the nonlinear fitting tool in MATLAB, the experimental data is fitted into the model of the series RLC circuit using equation (7.9). The fitted RLC circuit parameters are presented in Table 7.1. While imperfect, the fitting curve captures the general electrical behavior of the circuit, showing that the circuit is more complex than a simple RLC circuit with a variable capacitance. Because the model is close enough to the circuit response, it is the model considered for the rest of this article. Also, FIG.7.3 shows the experimental time-response of different input signals and output signals used to obtain the gain of the circuit.

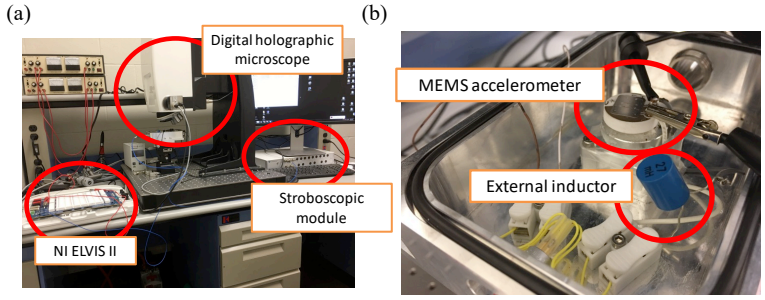


Figure 7.2 Experimental setup: (a) circuit connection showing the accelerometer and the external inductor. (b) equipment used for measurements: data acquisition system, DHM, and stroboscopic module.

Table 7.1: MEMS circuit parameters extracted from fig.7.3

Parameter	Definition	Value
L_e	Total circuit series inductance	25.66 mH
R_e	Total circuit series resistance	799.2 Ω
C_0	Nominal MEMS capacitance	10 pF
C_p	Circuit parasitic capacitance	224.74 pF

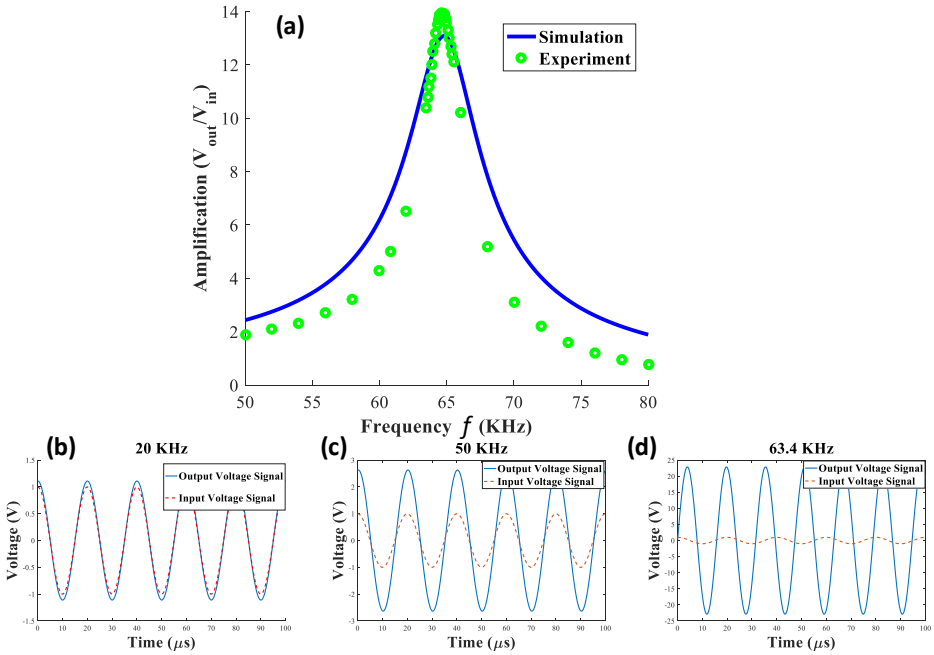


Figure 7.3 Electrical circuit identification plots including: (a) Circuit frequency response showing the amplification (gain) of the voltage across the capacitance with respect to the input voltage. Resonance was detected at f_e = 64.6 kHz with 13 times gain. b-d: time history response at different input frequency values.

7.2.2. Response Comparison: Mechanical Resonance, Electrical Resonance, Double Resonance

Next, we compare the response of the MEMS accelerometer to mechanical resonance only, to electrical resonance only, and to double resonance. The ambient pressure was reduced to around *20 Pa* to overcome the high viscous damping and the accelerometer was excited near its mechanical resonance, using the *2V* AC-amplitude signal of *190 Hz*. Consequently, the amplitude of the out-of-plane deflection reaches around 20 nm as shown in FIG.7.4,a. Afterwards, we stimulated excited the accelerometer by using a *2V* AC-amplitude signal of *60.8 kHz* near the electrical resonance frequency. We expected the voltage of the resonator to be amplified to up to five times the input voltage in FIG.7.3. However, the MEMS device attenuates the effect of the voltage amplification and thus, reduces the deflection to *10nm* as the AC excitation frequency gets far from the mechanical resonance frequency (FIG.7.4,b). Therefore, to trigger the electrical resonance and to amplify the voltage output, we need an enhanced signal that excites the high-frequency electrical

resonance and low-frequency mechanical frequency simultaneously. Using a signal with two appropriate frequency components (a beating signal): One component of the proposed AC excitation signal is chosen to be near the electrical resonance, while the absolute difference between the two frequency components should be near the mechanical resonance magnitude. We experimentally obtained this signal by driving the resonator with two AC signals of *60.8 kHz* and *60.61 kHz*. The resonator responds to the difference between the two signals (*190 Hz*), while the electrical circuit amplified the two input components. The use of the mixed-frequency signal results in a large vibration amplitude of *275 nm* (FIG.7.4,c). Therefore, we show that we can use a small AC actuation voltage to trigger a large mechanical deflection of the resonator while simultaneously amplifying the voltage across it through double resonance excitation.

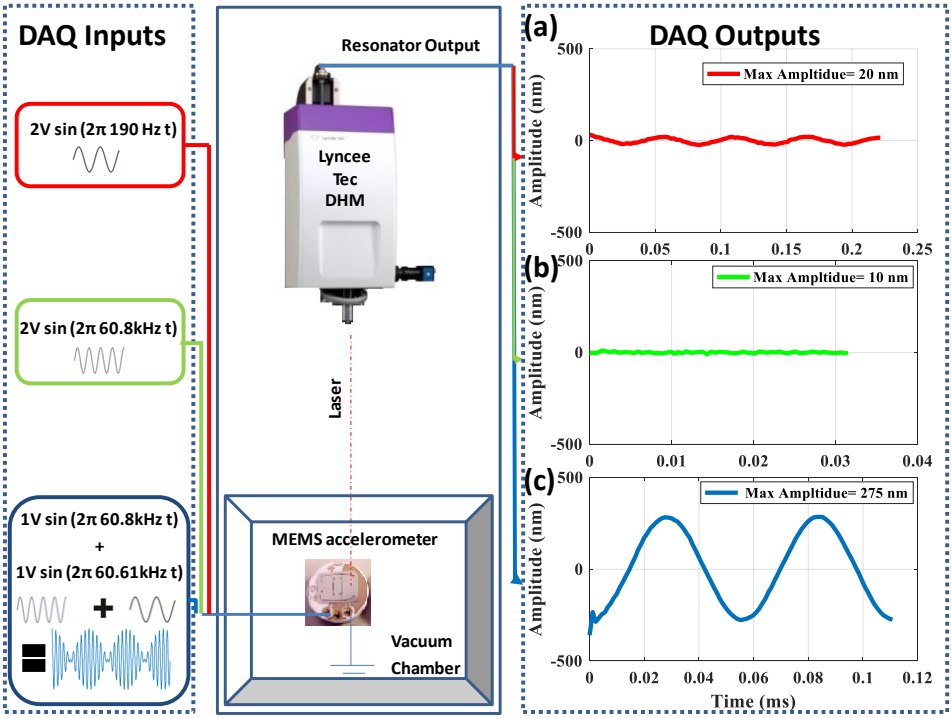


Figure 7.4 The experimental out-of-plane deflection of the resonator for three different cases: (a) close to the mechanical resonance frequency, a sinusoidal signal of 180 Hz and 2V of amplitude generates a deflection amplitude of 20 nm. (b) Relatively close to the electrical resonance of the circuit, a sinusoidal signal of 60.8 kHz and 2V of amplitude generates a deflection amplitude of ~10 nm. (c) A mixed signal composed of two frequencies: 60.8 kHz and 60.61 kHz, each with 1 V of amplitude, generates a deflection amplitude of 275 nm.

7.2.3. Double Resonance Frequency Response

As we have two frequency components, we perform frequency sweep by fixing one of the excitation frequencies (f_1) while sweeping the other frequency (f_2). Figure 7.5 shows experimentally and numerically the frequency response of the resonator when driven with $f_1 = 60.8\ kHz$ and $f_1 = 63.1\ kHz$, respectively, while sweeping f_2. Each frequency component has an amplitude of $1\ V_{AC}$. In both cases, the responses of the resonator are compared to the response a single sinusoidal input force around the mechanical resonance with an amplitude of $2V$ as a reference. The pure AC signal is swept from $85\ Hz$ to $107.5\ Hz$, where the input frequency is halved for a pure AC signal because of the frequency doubling when using one AC source.

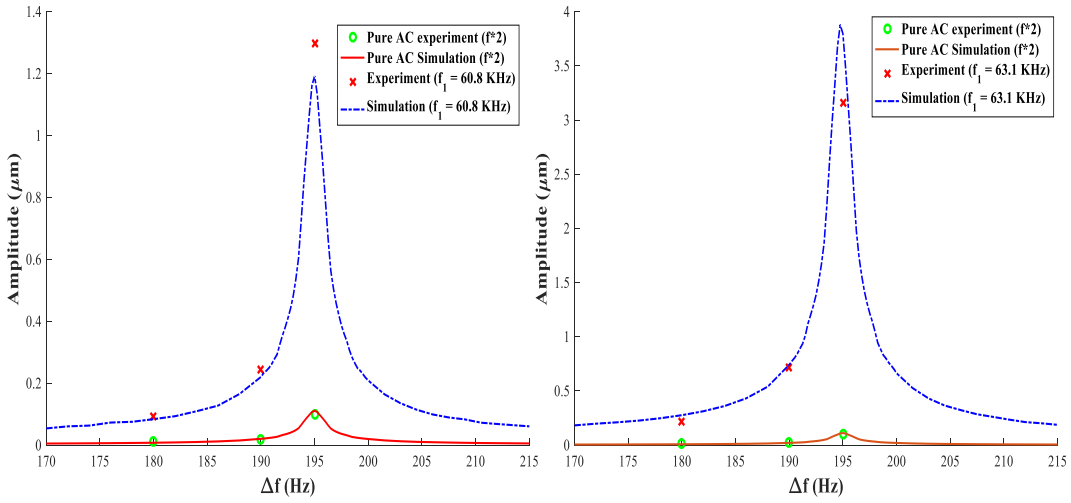


Figure 7.5 Double resonance excitation using a beating signal composed of two voltage sources each with an amplitude of 1V: (a) A frequency of f_1=60.8 kHz, and (b) A frequency of f_1=63.1 kHz and f_2 is swept both cases. Experimental results are shown with crosses and simulation is shown by a dashed line. Both cases are compared to the experimental and numerical simulation frequency response when excited classically by a single AC source at a frequency range of (85 Hz – 107.5 Hz) at 2V, that has a maximum amplitude of 0.1 μm.

While driven by a single AC source, the resonator has a maximum amplitude of $0.2\ \mu m$ at $97.5\ Hz$. However, the resonator has a higher maximum amplitude of $1.3\ \mu m$ with a 13 times amplification (Fig.7.5,a) and $3.16\ \mu m$ at $195\ Hz$ with a 30 times amplification (Fig.7.5,b) while driven by a double resonance excitation . Higher amplitude is when $f_1 = 63.1\ kHz$ because it is closer to the electrical resonance frequency of the circuit compared to $f_1 = 60.8\ kHz$.

The experimental response is then compared to the simulation results obtained by solving (3.1) and (7.6) simultaneously, considering the capacitance change due to the MEMS deflection and taking into account squeeze film damping. The solution is obtained numerically using the Runge-Kutta method assuming zero initial conditions.

We note here that the fitted model of section 7.1 does not converge quite as well with the dynamical experimental data at frequencies around the mechanical resonance frequency. This might be because of the complicated motional components involved in the circuit or the additional parasitic in the circuit influencing the behavior of the RLC circuit more significantly at this point. Therefore, to create a better fit, the same nonlinear fitting program was used to perfectly fit an RLC circuit into the electrical frequency response of the electrical circuit in FIG.7.3 around the frequency of interest. For instance, the points around f=60.8 kHz were perfectly fit by an RLC-circuit model. The fit fails outside a small range, however. Utilizing this local fitting, we simulate the behavior of the resonator at 60.8 kHz using a fit and the behavior at 63.4 kHz using another fit. The simulated response, in either case, is shown in FIG.7.5 with dashed lines. The local fit appears to closely match the experimental data in both figures.

Next, we construct a three-dimensional plot of the mechanical response of the resonator as a function of the two excitation frequencies (f_1 and f_2), with frequency components of $V_{AC} = 1V$ each in FIG.7.6. However, we replace f_2 with Δf for the sake of clarity and to simplify the identification of the mechanical resonance frequency. The figure shows relative maxima around f_1=64.6 kHz and $\Delta f = 195\ Hz$, which correspond to the electrical resonance frequency and the mechanical resonance frequency, respectively. Regardless of the value of f_1, the electrical and the mechanical resonances can individually amplify the input signal even without interacting with each other. Moreover, a higher vibrational amplitude equals $8.9\ \mu m$ at $f_1 = 64.6\ kHz$ and $\Delta f = 195\ Hz$ which corresponds to the electrical circuit resonance and the mechanical resonance, respectively, showing the constructive addition of the effects of electric and mechanical amplification. The double resonance

actuation maintains the linear-vibrational-response of the resonator. This makes double-resonance excitation a powerful excitation method that can be used to actuate any MEMS or NEMS device with no regards to the internal design of the device and without changing the overall response behavior of the MEMS accelerometer.

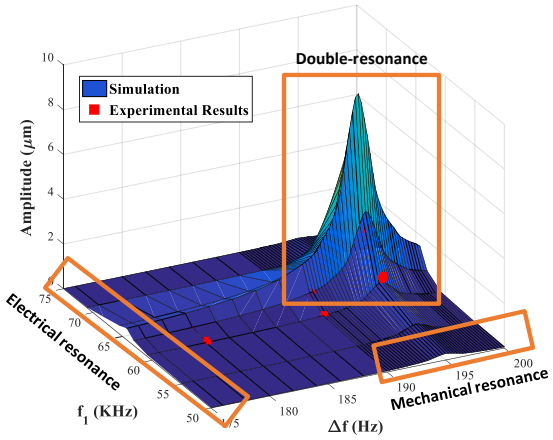


Figure 7.6 Three-dimensional plot of the simulated frequency response as a function of the excitation frequencies f_1 and ($f_2 = f_1 + \Delta f$). The voltage of each signal component is $V_1 = V_2 = 1$ V. The experimentally obtained points are in red.

Here, the existence of a large amplitude regime is formed by the intersection of the electrical resonance regime and the mechanical resonance regime. This high amplitude far exceeds the individual contribution of either resonance. This behavior can only be accessed by classical actuation means if the resonator is carefully designed to have an electrical resonance frequency close to the mechanical resonance frequency. While this imposes tight design tolerances, such as the need to greatly control the circuit parasitic, it remains possible. However, without changing the actual mechanical design, we showed using a double resonance to trigger the two primary resonance frequencies of the systems simultaneously. While this actuation method requires an additional voltage source, the actual voltage requirements are significantly lowered due to the large amplification of the signal.

The linear large vibrational amplitude is a notable achievement that is not possible for classical parallel-plate resonators actuation. This is explained by the interaction between the mechanical

deflection and the capacitance of the circuit inducing an inherent feedback in the system. As the maximum amplitude of vibration increases, the capacitance of the resonator changes, which shifts the electrical resonance frequency and attenuates the input voltage signal. This can theoretically reduce the risk of the pull-in voltage of the system. It should be noted that this effect depends on the parasitic capacitance of the circuit. When the ratio of the parasitic capacitance to the nominal capacitance is small, the system retains better stability at high amplification. However, as this ratio increases, the attenuation becomes less significant and the system starts behaving like the classical behavior but at significantly higher amplitude for a given input voltage.

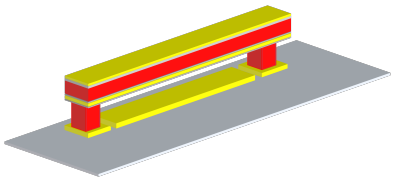
7.2. Double Resonance in Small MEMS Structures

In this section, a small, micro-fabricated MEMS structure, driven using double resonance excitation, is studied. The size of the MEMS device allows double resonance activation using a single AC source when a sufficiently large inductance is used. Alternatively, the multi-frequency excitation signal discussed in section 7.1 can be used.

The fabricated clamped-clamped microbeam resonator, is shown in FIG.7.7,a. The microbeam is fabricated on a silicon wafer coated with 500 nm of thermally grown silicon dioxide (SiO_2) layer. The lower electrode is formed by pattering the Cr/Au layer that is sputtered on the wafer surface. The microbeam is composed of a 4.2 μm polyimide layer coated from top with 50/200 nm Cr/Au layer. This layer is used to define the beam dimensions and act as hard mask to protect the beam during the reactive ion etching (RIE). The upper electrode is formed by coating the beam from bottom with 50/200/50nm of Cr/Au/Cr. The Cr is used to enhance the adhesion of the polyimide layer with other materials. The two electrodes are separated by 3.3 μm amorphous silicon layer. This layer is etched at the final stage of the fabrication to define the air gap.

(a)

Color	Material	Thickness	Modulus Of Elasticity	Density
	Silicon dioxide (SiO_2)	500 nm	85 GPa	2300 (Kg/m³)
	Chrome (Cr)	50 nm	279 GPa	7190 (Kg/m³)
	Gold (Au)	200 nm	79 GPa	19300 (Kg/m³)
	Amorphous silicon	3.3 µm	80 GPa	2330 (Kg/m³)
	Polyimide (Pi)	4.3 µm	8.5 GPa	1400 (Kg/m³)



(b)

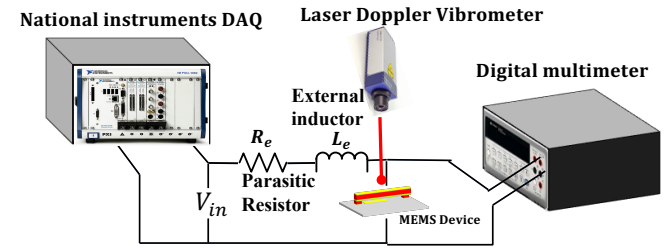


Figure 7.7 (a) A schematic of the clamped-clamped microbeam resonator, and a table showing different materials used for fabrication and their properties. (b) The measurement circuit showing the MEMS resonator, a voltage source (DAQ), external inductor, parasitic resistor, and measurement devices (Laser Doppler Vibrometer for the measurement of the mechanical response, and a digital multi-meter to record the voltage across the MEMS capacitor for the electrical characterization).

We characterized the mechanical and electrical properties of the MEMS resonator by studying the frequency response near the mechanical and electrical resonances. The mechanical resonance frequency of the fundamental mode was measured to be around 123 kHz using white noise signal excitation. The MEMS device is electrically modeled as a nominal parasitic capacitance C_p, formed by the deformable MEMS microbeam and the substrate beneath it, connected in parallel to a series branch of motional resistance, motional capacitance, and motional inductance, donated by R_M, C_{MEMS} and L_M , respectively. We create a resonant RLC circuit drive by connecting the MEMS device to a variable external inductance $R_{external}$, which results in a total resistance equal to R_e, Fig.7.7(b). The circuit's total resistance R_e is the equivalent parasitic resistance from the wires and the internal resistance of the inductor L_e . In our experiments, we varied L_e to control the series RLC resonance frequency and the voltage gain across the MEMS capacitor C_{MEMS}. The frequency sweep for electrical resonance characterization was conducted at ambient pressure to minimize the

effects of the MEMS deflection on the electrical parameters of the MEMS circuit, by introducing a high squeeze-film damping.

The frequency response for characterizing the pure electrical resonance is conducted by an impedance analyzer, as shown in Fig.7.8. We identified electrical resonance by one of the two methods: (i) by monitoring the amplitude of the circuit conductance *G* with respect to the frequency, Fig.7.8,a, where the electrical resonance corresponds to a global conductance maximum, and (ii) by monitoring the voltage across the MEMS capacitor C_{MEMS}, where the maximum voltage is achieved at the electrical resonance frequency as shown in Fig. 2(b). The increase in the conductance is due to the circuit reactance X_e going to zero at resonance, as shown in (7.13) and (7.14):

$$X_e = 2\pi f L_e - 1/(2\pi f C_{tot}) \tag{7.13}$$

$$G = R_e/(R_e^2 + X_e^2) \tag{7.14}$$

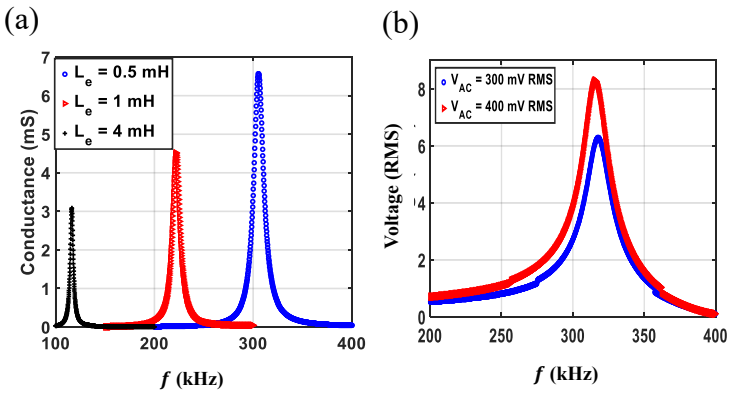


Figure 7.8 (a) Variation of electrical resonance frequency with respect to inductance. For higher inductance values the total resistance value increases, hence, a drop in the conductance value is expected. (b) Frequency response of the voltage across the MEMS device (with L_e = 0.5mH) that shows voltage amplification for two different small input voltages of 300mV and 400mV.

Next, we compare the response of the MEMS device with simultaneous activation of electrical and mechanical resonances to that actuated using conventional mechanical resonance alone. Fig.7.9,a shows the response of the device operated at atmospheric pressure, with V_{DC} = 30V and for various

AC voltages. We note that for this experiment, we have disconnected L_e. Fig.7.9,b shows the response of the double resonance driven circuit at the same pressure but with V_{DC} = 10V. In order to activate double resonance, the electrical resonance frequency was tuned to be near the mechanical resonance frequency by using L_e = 4 mH. To compare the two actuation methods, we find the product $V_{AC}V_{DC}$ that results in similar maximum vibrational amplitudes. For instance, to achieve a response of 0.7 μm, the required product of $V_{AC}V_{DC}$ without electrical resonance is 750 V^2 compared to only 50 V^2 when both resonances are simultaneously activated. Thus, as the DC component of the signal remains unamplified, we show an effective 15 times voltage amplification of the AC actuation signal by driving the MEMS resonator with double resonance drive.

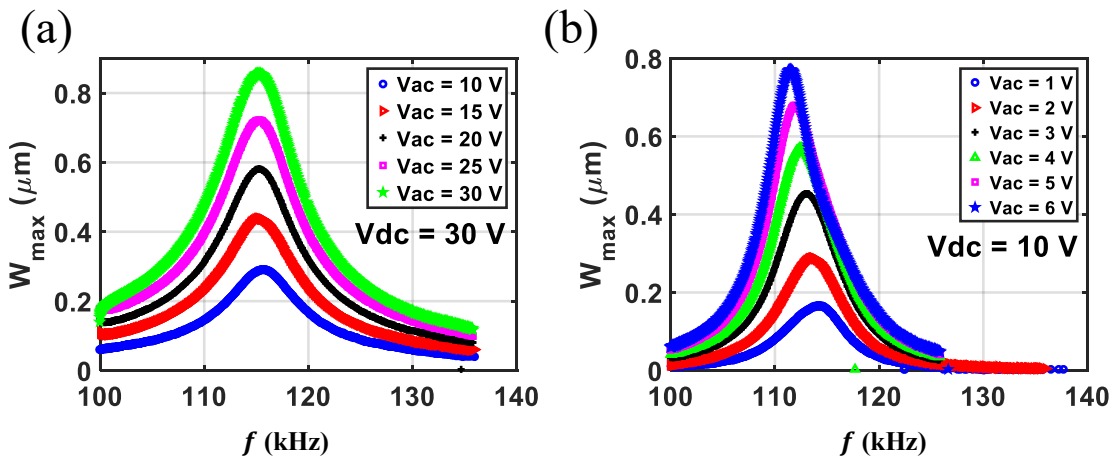


Figure 7.9 The frequency response of the MEMS resonator at atmospheric pressure when driven with: (a) mechanical resonance alone for V_{DC} = 30V (no external inductance was used in this experiment). (b) Double resonance drive with V_{DC} = 10V and L_e = 4 mH (electrical resonance = 116 kHz).

While double resonance activation was simple to achieve for the experimental results shown in Fig. 7.9,b, due to the proximity of the systems' mechanical and electrical resonance frequencies, this might not be the case for general MEMS devices. To overcome this limitation, we introduce a multi-frequency excitation signal composed of a beating signal with two frequency components: f_1 and f_2. Note that we do not use any DC bias for this experiment. Due to the quadratic voltage term shown in equation (2.2), the frequency spectrum of the resulting electrostatic force include $2f_1$, $2f_2$,

(f_1+f_2), and (f_1-f_2) frequency components. Therefore, for double resonance activation, either f_1 or f_2 is selected to be around the electrical resonance frequency while at least one of the forcing spectral components is made to be equal to the mechanical resonance frequency. To demonstrate this concept, we show in Fig.7.10, the response of the MEMS resonator to a multi-frequency input signal with electrical resonance frequency at 308 kHz and a mechanical resonance frequency at 123 kHz (a mismatch of 185 kHz). We note that this experiment was also conducted under atmospheric pressure. In Fig.7.10,a, the input signal has a fixed frequency component, f_1 = 308 kHz at different V_{AC1} values, near the electrical resonance frequency, while the second frequency component f_2, was swept such that (f_1-f_2) is near the mechanical resonance (123 kHz). This resulted in voltage amplification of V_{AC1} across the MEMS resonator (capacitor) due to the electrical resonance and overall forcing amplification, hence, higher amplitude of motion.

In contrast, when both f_1 and f_2 are far from the electrical resonance frequency, significantly higher input voltages are required to achieve comparable results, as shown in Fig.7.10,b. For instance, to achieve an amplitude of 0.36 μm, the product of $V_{AC1}V_{AC2}$ is 1176 V^2 while the required $V_{AC1}V_{AC2}$ with double resonance drive is about 39 V^2. Thus, Fig.7.10,b demonstrates a voltage amplification gain of ~30 through double resonance excitation. This amplification is almost twice the amplification obtained in Fig.7.10,b by matching the resonator electrical resonance to its mechanical resonance using a larger inductor. We attribute this higher gain to the flexibility of using a smaller external inductor, and hence less parasitic resistance in the circuit, when matching the two frequencies is not required. Finally, to demonstrate the increase in the quality factor of the system, we compare the response of the MEMS device with and without double resonance activation, Fig.7.10,c. Each case utilizes a multi-frequency signal excitation such that f_1 and f_2 produce a forcing signal that has a frequency-spectrum component equal to the mechanical resonance frequency. A significant increase in the quality factor of the MEMS resonator and a

vibrational amplitude amplification of ~30 times was found when the system was driven using double resonance drive.

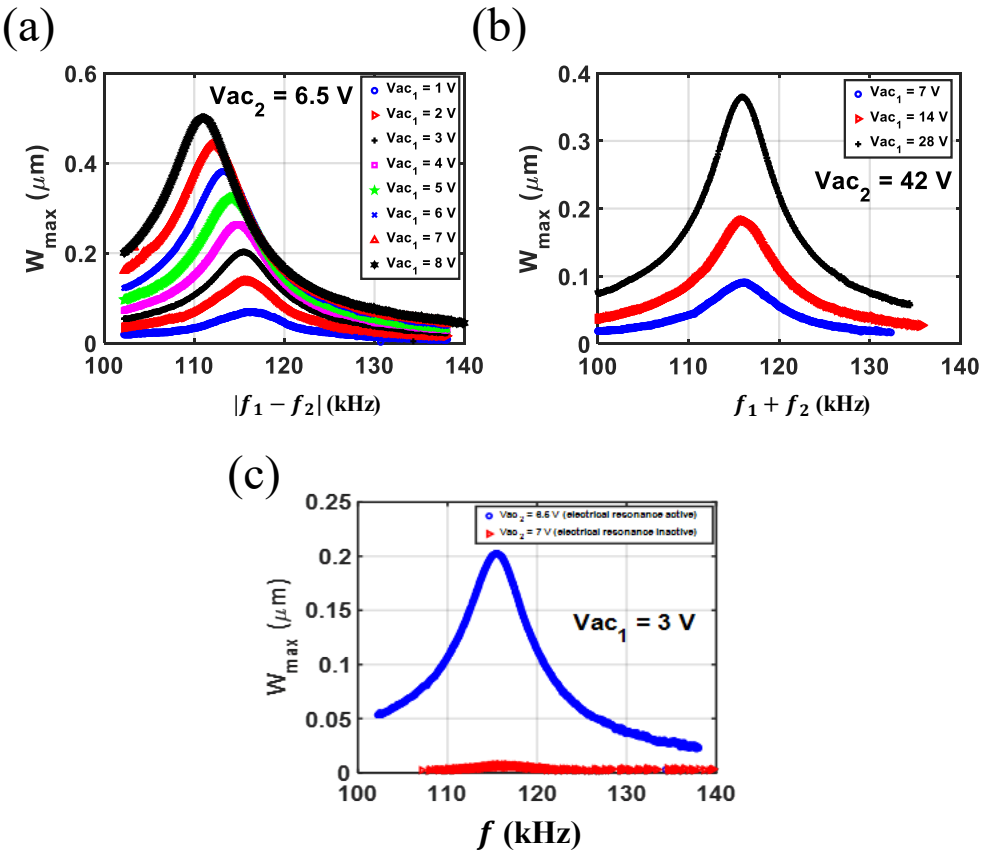


Figure 7.10 Frequency response of the MEMS device for two cases. (a) Double resonance excitation: f_1 is fixed at the electrical resonance (308 kHz), V_{AC2} = 6.5 V and f_2 is swept such that $|f_1-f_2|$ is around the mechanical resonance. We show multiple values of V_{AC1} in this figure. (b) Mixed frequency excitation away from the electrical resonance frequency: f_1 is chosen arbitrarily far away from the electrical and mechanical resonance frequency (80 kHz) with V_{AC2}=42 V while f_2 was swept such that f_1+f_2 is around the mechanical resonance frequency. We show multiple values of V_{AC1} , however, we show that to achieve similar deflection in (a), significantly more voltage is required. (c) Shows the increase in the quality factor when the resonator is operated using double resonance (blue circle, V_{AC1}= 3V, V_{AC2}=6.5V, f_1=308 kHz, f_2 = 170 to 210 kHz, $f_{effective}=f_1-f_2$) compared with the case when electrical resonance is not active (red triangle, V_{AC1}=3V, V_{AC2}= 7 V, f_1=80 kHz, f_2= 20 to 60 kHz, $f_{effective} = f_1+f_2$). Here, $f_{effective}$ is a frequency near the MEMS mechanical resonance frequency.

7.3. Response Modification for Computing

Another possible advantage of electrical resonance actuation, even in case of a frequency mismatch, is the ability to amplify the static deflection of MEMS devices, as previously demonstrated previously by [102] [103]. This is because of the quadratic relationship between the input voltage

forcing and the electrostatic forcing. Thus, the electrostatic actuation force using a single AC signal with a frequency $f_e \gg f_m$ can be approximated as:

$$F_e = \frac{\varepsilon b\left(\left[0.5V_{act}^2 + 0.5V_{act}^2 Cos(4\pi f_e t)\right]\right)}{2(d-x)^2\left[\left((2\pi f_e)R_e C_{tot}\right)^2 + (1-(2\pi f_e)^2 L_e C_{tot})^2\right]} \tag{7.15}$$

As the MEMS device attenuates signals with frequencies that far exceed its resonance frequency, the term $0.5V_{act}^2 Cos(4\pi f_e t)$ in the numerator is negligible. Thus, the forcing in (17.5) is static with an amplified amplitude. We note here that the gain attained using this approach is not constant as the total capacitance of the system depends on the MEMS deflection. The gain is highly variable when $C_{tot} \cong C_{MEMS}$ since the MEMS deflection will shift the electrical resonance frequency of the system away from the supply signal frequency, which is typically constant. This shift would reduce the gain of the system. This interaction can be seen as an internal feedback in the system. Interestingly, increasing the parasitic capacitance of the MEMS system tends to eliminate this internal feedback effect if the parasitic capacitance far exceeds the MEMS variable capacitance. In this case, the capacitance change due to the MEMS deflection will have a negligible effect on the overall capacitance of the system, which leads to a nearly constant electrical resonance frequency during operation.

The inherent internal feedback of MEMS devices with a low parasitic capacitance is very useful. Theoretically, it can allow the MEMS device to operate far beyond the traditional stability regime, significantly reducing the risk of pull-in. A very large parasitic capacitance leads to the usual saddle node bifurcation response of the MEMS device, which leads to the pull-in instability at deflections larger than $1/3$ of the gap between the electrodes. Interestingly, for some moderate C_p value, a response bistable similar to that of a MEMS arch around the snapthrough regime. At that point, the parasitic capacitance is large enough to case instability in the intermediate section of the bifurcation diagram (FIG.7.11,a), which separates the previously single stable branch into two branches,

creating bistability. This behavior is very useful for computing applications, such as using a network of MEMS devices as a CTRNN, as was demonstrated in chapter 4.

Figure 7.11,b theoretically demonstrates the effects of parasitic capacitance on the response of the electrical resonance-driven MEMS clamped-clamped beam from section 7.2 using the following input signal parameters $f_1 = 116\ kHz$, $f_{electrical} = 308\ kHz$ and $V_{DC2} = V_{AC2} = 0\ Volt$. The voltage V_{AC1} was swept to show the effects of internal negative feedback at higher deflection for different ratios between the paratactic capacitance and the MEMS nominal capacitance C_p/C_{MEMS}values. As anticipated, increasing this ratio reduces the internal negative feedback and increases the deflection of the MEMS device.

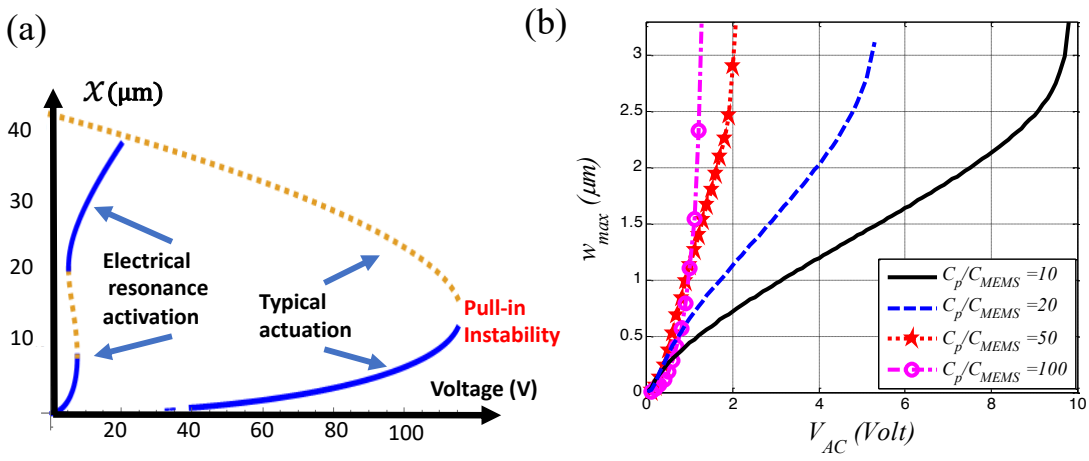


Figure 7.11 The use of electrical resonance as a means to enhance the static response of MEMS devices by increasing the MEMS deflection due to voltage amplification. The voltage amplification is more pronounced when C_p/C_{MEMS} is high. (a) A simulated response of a SDOF MEMS device from section 7.1 showing bistability at moderate C_p/C_{MEMS} values. (b) A simulated response of a MEMS microbeam from section 7.2 showing large response amplification at high C_p/C_{MEMS} due to the lack of internal feedback. The response amplitude is lower when the parasitic capacitance smaller (black line) due to the presence of internal feedback.

7.4. Conclusions

In conclusion, this chapter a passive means of signal amplification to enhance the MEMS actuation signal by utilizing electrical resonance. Two different schemes are demonstrated to utilize this signal to improve the dynamical response of MEMS resonators. The first approach relies on tuning the electrical resonance frequency to coincide with the mechanical resonance frequency, using a

variable external inductor, which is viable when the two resonances are proximate. A more generic approach to activate electrical and mechanical resonances simultaneously is also shown by actuating the device using a multi-frequency signal. In this case, one of the signal's frequency components was near the electrical resonance. The other component was chosen such that at least one of the forcing spectral components (such as f_1+f_2 or f_1-f_2) matches the MEMS mechanical resonance. In both the cases, a high amplitude response was recorded. An increase in the quality factor of the resonator response was also shown. We note that the activation of simultaneous electrical and mechanical resonances does not require any changes in the design of MEMS devices. However, the electrical resonance frequency and its corresponding voltage gain may differ between similar devices due to parasitic capacitance and resistance variation. More precise fabrication and tuning of external electrical components (inductor, capacitor) can be used to alleviate this issue. In addition, one can tune the series-connected external inductor or use an external variable capacitor in parallel with the MEMS device to tune the electrical resonance, if needed. Nonetheless, the simultaneous electrical and mechanical resonance activation scheme demonstrated here may alleviate the need for CMOS incompatible high AC voltage source or amplifiers for actuating these devices, especially where high AC input signal is necessary, such as, nonlinear operation of M/NEMS actuation and/or operation at moderate to atmospheric pressure.

The presented approach can also be utilized to extend the operational range of MEMS devices by introducing internal negative feedback, assuming a low parasitic capacitance. More importantly for this dissertation, the proposed approach is very appealing for MEMS computing applications as operation around electrical resonance with moderate parasitic circuit parasitic capacitance generates a bistable behavior in the MEMS device that can be used to emulate a continuous-time recurrent neuron (CTRN). Moreover, since the voltage signal is amplified through electrical resonance, the MEMS neuron will require up to an order of magnitude less voltage to operate, eliminating one of the biggest MEMS computing challenges.

CHAPTER 8

FUTURE RESEARCH DIRECTIONS

This thesis demonstrates the use of MEMS devices for colocalized sensing and computing both in the CTRNN architecture and the RC architecture. This section discusses potential future directions stemming from the presented results.

8.1. Investigating MEMS Parameter Space

A major obstacle in the reservoir computing field is the vague definition of the RC requirements and the lack of dynamical understanding of the systems proposed for computing. Consequently, researchers are forced to analyze the utilized dynamical system as a black box, which is optimized by scanning the parameter space – by trial and error. This limitation also resulted in the lack of baseline for comparing various RC systems.

Bifurcation analysis can be used as a starting step to address this problem. Using this approach, the dynamical complexity of the MEMS system can be assessed in order to qualitatively reach a suitable dynamical regime. The MEMS dynamics may be tuned by applying a positive delayed feedback to reach an overall effective regime of operation for a fixed input voltage. The edge of chaos is said to be ideal for computation. More specifically, the edge of chaos is known for having maximal information. However, research show that retaining maximum information does not necessarily translate to optimal computational performance. Moreover, work involving the edge of chaos only considered autonomous dynamical system (with no inputs). RC and CTRNNs process information attained from their inputs. Thus, edge of chaos may be inappropriate. Hence, the regime of maximal performance may vary for different inputs (biasing signal and sensed signal). Thus, to reach optimize the response of the MEMS device, hyper-parameter dynamical study is required. The dynamical studies requirement may be investigated by utilizing some terminology from the field of Boolean networks by creating an approximation model to represent the average

response of the network with respect to the hyper-parameters. Furthermore, by studying the basin of attraction of the approximate model attractors, we may be able to visualize the capabilities of the dynamical system for input separation and for classification.

The computational power of the MEMS reservoir will be assessed using a Memory Capacity test. In this test, a piecewise constant input signal $u(t)$ is fed to the MEMS reservoir such that:

$$u(t) = a_j, (j-1)T \leq t < jt, k = 1, \dots, J \tag{8.1}$$

where a_j is the j^{th} value of the input signal and T is the duration of each input value. The reservoir is expected to retain memory of its previous inputs. i.e, recalling $u(t-kT)$. To identify the MEMS capabilities, the correlation between the MEMS RC output and the delayed input signal for a wide range of delays [105]:

$$MC = \sum_{k=1}^{K} R^2[u(t-kT), y(t)] \tag{8.2}$$

where MC is the memory capacity measure calculated based on delays $k = 1, \dots, K$ and $R^2[a, b]$ is the R^2 statistical measure between the random variables a and b. Memory capacity is widely used in the literature as a benchmark for computational performance.

Another measure of interest in this objective is the susceptibility to noise. Practical implementation of any computational method impels the RC system to retain some degree of noise-resistance. Indeed, this embodies the requirement R2 of RC. Thus, noise susceptibility test is both a test of real-life applicability and the fitness of the MEMS device to perform RC computation. To this end, noise will be introduced to the reservoir input signals, the reservoir mask and/or the reservoir feedback to address most possible points of noise injection in practice.

8.2. MEMS Response Tuning via Nonlinear Operation

A notable benefit for utilizing MEMS devices for simultaneous sensing and computing is the abundance of methods to induce nonlinearities. Nonlinearities like veering and internal resonance

can facilitate differing degrees of interaction between the MEMS modeshapes, allowing response flexibility and additional nonlinear richness. Other nonlinearities induced from alternative excitation schemes can be used to amplify the MEMS response and provide additional complexity (such as excitation based on a moving boundary), other schemes can eliminate the pull-in instability while retaining dynamical complexity (by utilizing repulsive electrostatic forcing due to fringe fields).

A possible future direction is to model the influence of these various instabilities on the response of a MEMS virtual RC ensemble based on the scheme developed in this dissertation. Additionally, nonlinearities may be added as means to encode additional environmental conditions (such as temperature and pressure, in addition to acceleration), thus, allowing the developed MDOF MEMS RC to perform both as an ensemble computing unit and an ensemble sensing unit, to create intelligent, specialized sensing and computing units.

Finally, there may be merit to studying the MEMS response using nonlinear excitation schemes. For instance, using a moving boundary allows for the possibility of either dynamically changing the MEMS stiffness and natural frequency based on the position of the boundary, allowing a flexible tuning of the MEMS response to improve the computational performance of the RC for a larger bandwidth.

8.3. MEMS Colocalized Sensing-and-Computing in Unconventional Robotics

Unconventional robotics is an emerging field of research focusing, among other things, on soft and micro-robotics. Robots in this field require intricate control as they often experience complex mechanical deformations or operate in complex-to-model fluidic environments. Due to power and size constraints in these systems, classical control schemes are typically inappropriate to use. The developed colocalized sensing-and-computing MEMS RC scheme in this dissertation, however,

may prove to be useful for such robotic systems as it requires minimal analog components in addition to the MEMS sensor.

A possible future research may involve simulating and train a soft- or micro-robot equipped with a MEMS accelerometer RC to follow a predefined trajectory or maintain stability. This research direction is similar to the active CTRNN categorical preceptor problem conceptually. However, further complexities are introduced in this case by considering the dynamics of the robot. Ultimately, MEMS RC may be introduced into the body of the non-conventional robot as a smart robotic sensor, enabling true morphological computing [106].

8.4. Local Biomonitoring via Printable Smart Tattoos

Wearable devices promise great improvement to public health. However, due to a combination of inconvenience to use and expense, they have yet to live up to their potential. Printable and flexible electronic technologies may enable the fabrication of inexpensive, truly wearable electronics for biomonitoring. The battery life of these devices may be significantly enhanced by the use of the MEMS computing schemes introduced in this dissertation, by eliminating, or significantly reducing the need, for cloud computing. Thus, theoretical and experimental research into the use of colocalized MEMS sensing-and-computing schemes appears to be a natural extension to this dissertation.

CHAPTER 9

CONCLUSION

In this dissertation, we presented multiple methods to use MEMS devices for colocalized sensing and computing. First of which is using networks of MEMS devices to emulate continuous-time recurrent neural networks (CTRNNs). To this end, we identify the dynamical properties necessary for each MEMS device to emulate an individual CTRN. Pull-in, snapthrough and electrical resonance were shown to satisfy these conditions. Reservoir computing (RC) is the second scheme presented in this dissertation, in which a single MEMS device doubles as a sensor and a reservoir of virtual neurons. MEMS devices were found to be particularly useful as continuous structures as they introduce additional complexity through modal interactions. Additionally, we propose a novel MEMS actuation method to significantly reduce the voltage required to drive MEMS devices in order to facilitate the use of MEMS devices in their nonlinear regimes. Finally, we concluded this dissertation by discussing future research directions that may stem from this work.

Appendix A: Composite Microbeam Analysis

To simplify the study of our composite microbeam, we utilize an equivalent beam model to represent our microbeam as a single layer beam using the equivalent area method. The neutral axis of the single-layer equivalent beam was attained by considering a beam with the same first moment of area as that of the composite beam. Thus, the position of the single layer microbeam, $\bar{y}$, is obtained using (a1):

$$\bar{y}\sum_{i=1}^{6} A_i = \sum_{i=1}^{6} A_i y_i \tag{a1}$$

where y_i represents the location of the neutral axis of layer *i* along the y-axis, and A_i is the cross-sectional area of layer *i* calculated by adjusting the width of every layer such that $b_{i,eff} = b_i E_i / E_1$. See Fig.A1.

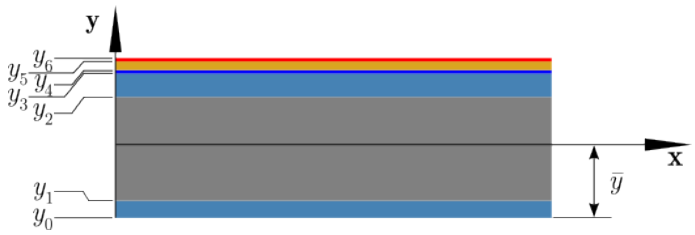


Figure A1 Neutral axis of the multi-layered microbeam

After calculating the neutral axis position, it is possible to find the effective modulus of elasticity of the single layer microbeam, $I_{c,eff}$ using the parallel axis theorem on the microbeam transformed section:

$$I_{c,eff} = \sum_{i=1}^{6} I_{c,i} \tag{a2}$$

Where $I_{c,i}$ is the second moment of inertia of layer *i* that can be computed as follows:

$$I_{c,1} = \frac{b_1 h_1^3}{12} + b_1 h_1 \left(\frac{h_1}{2} - \bar{y}\right)^2 \tag{a3}$$

$$I_{c,i} = \frac{b_i h_i^3}{12} + b_i h_i \sum_{j=1, j<i} \left(h_j + \frac{h_i}{2} - \bar{y}\right)^2 \tag{a4}$$

where h_i and b_i are the thickness and width of layer i, respectively. Finally, the effective flexural rigidity, $(EI_c)_{eff}$, and mass per unit length, $(\rho A_{cs})_{eff}$ can be attained by simply summing the flexural rigidity and mass per unit length of all layer in the composite microbeam:

$$\begin{cases} (EI_c)_{eff} = \sum_{i=1}^{6} E_i I_{c,i} \\ (\rho A_{cs})_{eff} = \sum_{i=1}^{6} \rho_i A_{cs,i} \end{cases} \tag{a5}$$

References

[1] W. A. Daxwanger and G. K. & Schmidt, "Skill-based visual parking control using neural and fuzzy networks," *Proc. IEEE SMC,* vol. 2, pp. 1659-1664, Oct 1995.

[2] H. M. Kim, J. Dickerson and B. & Kosko, "Fuzzy throttle and brake control for platoons of smart cars," *Fuzzy Sets and Systems,* vol. 84, no. 3, pp. 209-234, 1996.

[3] T. Sorsa, H. N. Koivo and H. & Koivisto, "Neural networks in process fault diagnosis," *IEEE Transactions on systems,* vol. 21, no. 4, pp. 815-825, 1991.

[4] Z. Du, B. Fan, X. Jin and J. & Chi, "Fault detection and diagnosis for buildings and HVAC systems using combined neural networks and subtractive clustering analysis," *Building and Environment,* vol. 73, pp. 1-11, 2014.

[5] F. M. Alsaleem, R. Abiprojo and &. J. Arensmeier, "HVAC system cloud based diagnostics model," in *Proc. 15th Int. Refrig. Air Conditioning Conf*, Purdue. West Lafayette, IN, USA, 2014.

[6] E. Sazonov, Wearable Sensors: Fundamentals, implementation and applications, Elsevier, 2014.

[7] A. L. Alfeo, P. Barsocchi, M. G. Cimino, D. La Rosa, F. Palumbo and G. & Vaglini, "Sleep behavior assessment via smartwatch and stigmergic receptive fields," *Personal and ubiquitous computing,* vol. 22, no. 2, pp. 227-243, 2018.

[8] B. Ibrahim and R. & Jafari, "Continuous blood pressure monitoring using wrist-worn bio-impedance sensors with wet electrodes," in *IEEE Biomedical Circuits and Systems Conference*, 2018.

[9] E. Markvicka, G. Wang, Y. C. Lee, G. Laput, C. Majidi and L. & Yao, "ElectroDermis: Fully Untethered, Stretchable, and Highly-Customizable Electronic Bandages," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019.

[10] M. D. Bartlett, E. J. Markvicka and C. & Majidi, "Rapid fabrication of soft, multilayered electronics for wearable biomonitoring," *Advanced Functional Materials,* vol. 26, no. 46, pp. 8496-8504, 2016.

[11] M. M. Waldrop, "The chips are down for Moore's law," *Nature News,* vol. 530, no. 7589, p. 144, 2016.

[12] E. Pop, S. Sinha and K. E. & Goodson, "Heat generation and transport in nanometer-scale transistors," *Proceedings of the IEEE,* vol. 94, no. 8, pp. 1587-1601, 2006.

[13] W. Haensch, Nowak, E.J., Dennard, S. R.H., B. P.M., D. A., K. O.H., W. X. A., J. Johnson and M. & Fischetti, "Silicon CMOS devices beyond scaling," *IBM Journal of Research and Development,* vol. 50, no. 4.5, pp. 339-361, 2006.

[14] W. Shi and S. & Dustdar, "The promise of edge computing," *Computer,* vol. 49, no. 5, pp. 78-81, 2016.

[15] H. Kalantarian, C. Sideris, B. Mortazavi, N. Alshurafa and M. & Sarrafzadeh, "Dynamic computation offloading for low-power wearable health monitoring systems," *IEEE Transactions on Biomedical Engineering,* vol. 64, no. 3, pp. 621-628, 2016.

[16] J. Zhou, Z. Cao, X. Dong and X. & Lin, " Security and privacy in cloud-assisted wireless wearable communications: Challenges, solutions, and future directions," *IEEE wireless Communications,* vol. 22, no. 2, pp. 136-144, 2015.

[17] C. D. Schuman, T. E. Potok, R. M. Patton, J. D. Birdwell, M. E. Dean, G. S. Rose and J. S. & Plank, "A survey of neuromorphic computing and neural networks in hardware," *arXiv preprint arXiv:1705.06963,* 2017.

[18] P. Hasler and &. L. Akers, "Vlsi neural systems and circuits," in *Computers and Communications*, Phoenix, 1990.

[19] L. Tarassenko, M. Brownlow, G. Marshall, J. Tombs and A. & Murray, "Real-time autonomous robot navigation using VLSI neural networks," *Advances in neural information processing systems,* pp. 422-428, 1991.

[20] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE,* vol. 78, no. 10, pp. 1629-1636, 1990.

[21] E. M. Green, R. van Mourik, C. Wolfus, S. B. Heitner, O. Dur and M. J. & Semigran, " Machine learning detection of obstructive hypertrophic cardiomyopathy using a wearable biosensor," *npj Digital Medicine,* vol. 2, no. 1, pp. 1-4, 2019.

[22] P. Pierleoni, A. Belli, L. Palma, M. Pellegrini, L. Pernini and S. & Valenti, " A high reliability wearable device for elderly fall detection," *IEEE Sensors Journal,* vol. 15, no. 8, pp. 4544-4553, 2015.

[23] M. Bachlin, M. Plotnik, D. Roggen, I. Maidan, J. M. Hausdorff, N. Giladi and G. & Troster, "Wearable assistant for Parkinson's disease patients with the freezing of gait symptom," *IEEE Transactions on Information Technology in Biomedicine,* vol. 14, no. 2, pp. 436-446, 2009.

[24] E. M. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE transactions on neural networks,* vol. 15, no. 5, pp. 1063-1070, 2004.

[25] A. L. Hodgkin and A. F. & Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology,* vol. 117, no. 4, p. 500, 1952.

[26] E. M. Izhikevich, " Simple model of spiking neurons," *IEEE Transactions on neural networks,* vol. 14, no. 6, pp. 1569-1572, 2003.

[27] Y. C. Cheng, W. M. Qi and W. Y. & Cai, "Dynamic properties of Elman and modified Elman neural network," in *International Conference on Machine Learning and Cybernetics*, 2002.

[28] S. Hochreiter and J. & Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[29] B. Yamauchi and R. & Beer, " Integrating reactive, sequential, and learning behavior using dynamical neural networks," *From Animals to Animats,* vol. 3, pp. 382-391, 1994.

[30] A. N. Tait, T. F. De Lima, E. Zhou, A. X. Wu, M. A. Nahmias, B. J. Shastri and P. R. & Prucnal, " Neuromorphic photonic networks using silicon photonic weight banks," *Scientific reports,* vol. 7, no. 1, pp. 1-10, 2017.

[31] S. A. Vigraham and J. C. & Gallagher, "Ctrnn-eh in silicon: challenges in realizing configurable ctrnns in vlsi," in *International Conference on Evolutionary Computation*, 2006.

[32] G. Schöner, Dynamic thinking: A primer on dynamic field theory, Oxford University Press, 2016.

[33] F. M. Alsaleem, M. H. Hasan and M. K. & Tesfay, "A MEMS nonlinear dynamic approach for neural computing.," *Journal of Microelectromechanical Systems,* vol. 27, no. 5, pp. 780-789, 2018.

[34] G. Tanaka, T. Yamane, J. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano and A. & Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks,* vol. 115, pp. 100-123, 2019.

[35] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman and S. & Massar, "Optoelectronic reservoir computing," *Scientific reports,* vol. 2, p. 287, 2012.

[36] L. Larger, M. Soriano, D. Brunner, L. Appeltant, J. Gutiérrez, L. Pesquera, C. Mirasso and I. & Fischer, "Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing," *Optics express,* vol. 20, no. 3, pp. 3214-3249, 2012.

[37] H. Jaeger, "The “echo state” approach to analysing and training recurrent neural networks-with an erratum note," German National Research Center for Information Technology GMD Technical Report, Bonn, Germany, 2001.

[38] W. Maass, T. Natschläger and H. & Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural computation,* vol. 14, no. 11, pp. 2531-2560, 2002.

[39] L. Appeltant, M. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. Mirasso and I. & Fischer, "Information processing using a single dynamical node as complex system," *Nature communications,* vol. 2, no. 1, pp. 1-6, 2011.

[40] J. D. Hart, L. Larger, T. E. Murphy and R. & Roy, "Delayed dynamical systems: networks, chimeras and reservoir computing," *Philosophical Transactions of the Royal Society A,* vol. 377, no. 2153, 2019.

[41] N. D. Haynes, M. C. Soriano, D. P. Rosin, I. Fischer and D. J. & Gauthier, "Reservoir computing with a single time-delay autonomous Boolean node," *Physical Review E,* vol. 91, no. 2, 2015.

[42] D. Brunner, B. Penkovsky, B. A. Marquez, M. Jacquot, I. Fischer and L. & Larger, "Tutorial: Photonic neural networks in delay systems," *Journal of Applied Physics,* vol. 124, no. 15, 2018.

[43] K. I. Kitayama, M. Notomi, M. Naruse, K. Inoue, S. Kawakami and A. & Uchida, "Novel frontier of photonics for data processing—Photonic accelerator," *APL Photonics,* vol. 4, no. 9, 2019.

[44] G. Dion, S. Mejaouri and J. & Sylvestre, "Reservoir computing with a single delay-coupled non-linear mechanical oscillator," *Journal of Applied Physics,* vol. 124, no. 15, 2018.

[45] B. Penkovsky, X. Porte, M. Jacquot, L. Larger and D. & Brunner, "Coupled nonlinear delay systems as deep convolutional neural networks," *Physical review letters,* vol. 123, no. 5, 2019.

[46] K. Nakajima, K. Fujii, M. Negoro, K. Mitarai and M. & Kitagawa, "Boosting computational power through spatial multiplexing in quantum reservoir computing," *Physical Review Applied,* vol. 11, no. 3, 2019.

[47] T. Dalgaty, E. Vianello, B. De Salvo and J. & Casas, "Insect-inspired neuromorphic computing," *Current opinion in insect science,* vol. 30, pp. 59-66, 2018.

[48] E. Arena, P. Arena, R. Strauss and L. & Patané, "Motor-skill learning in an insect inspired neuro-computational control system," *Frontiers in Neurorobotics,* vol. 11, p. 12, 2017.

[49] K. Nakajima, H. Hauser, T. Li and R. & Pfeifer, "Information processing via physical soft body," *Scientific reports,* vol. 5, 2015.

[50] B. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. Chandrasekaran, J. Bussat, R. Alvarez-Icaza, J. Arthur, P. Merolla and K. & Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE,* vol. 102, no. 5, pp. 699-716, 2014.

[51] N. Qiao and G. & Indiveri, "Analog circuits for mixed-signal neuromorphic computing architectures in 28 nm FD-SOI technology," in *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference*, 2017.

[52] T. Serrano-Gotarredona, T. Masquelier, T. Prodromakis, G. Indiveri and B. & Linares-Barranco, "STDP and STDP variations with memristors for spiking neuromorphic learning systems," *Frontiers in neuroscience,* vol. 2, p. 7, 2013.

[53] M. R. Azghadi, B. Linares-Barranco, D. Abbott and P. H. & Leong, " A hybrid CMOS-memristor neuromorphic synapse," *IEEE transactions on biomedical circuits and systems,* vol. 11, no. 2, pp. 434-445, 2016.

[54] T. Gokmen, M. J. Rasch and W. & Haensch, "Training LSTM networks with resistive cross-point devices," *Frontiers in neuroscience,* vol. 745, p. 12, 2018.

[55] K. Smagulova, K. Adam, O. Krestinskaya and A. P. & James, "Design of cmos-memristor circuits for lstm architecture," in *2018 IEEE international conference on electron devices and solid state circuits (EDSSC)*, 2018.

[56] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G. Nam and B. & Taba, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE transactions on computer-aided design of integrated circuits and systems,* vol. 34, no. 10, pp. 1537-1557, 2015.

[57] A. Vanarse, A. Osseiran and A. & Rassau, " A review of current neuromorphic approaches for vision, auditory, and olfactory sensors," *Frontiers in neuroscience,* vol. 10, p. 115, 2016.

[58] T. Delbruck and C. A. & Mead, "Adaptive photoreceptor with wide dynamic range," in *Proceedings of IEEE International Symposium on Circuits and Systems-ISCAS'94*, 1994.

[59] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco and T. & Delbruck, "Retinomorphic event-based vision sensors: bioinspired cameras with spiking output," *Proceedings of the IEEE,* vol. 102, no. 10, pp. 1470-1484, 2014.

[60] R. F. Lyon and C. & Mead, "An analog electronic cochlea. IEEE Transactions on Acoustics," *Speech, and Signal Processing,* vol. 36, no. 7, pp. 1119-1134, 1988.

[61] V. Chan, S. C. Liu and A. & van Schaik, "AER EAR: A matched silicon cochlea pair with address event representation interface," *IEEE Transactions on Circuits and Systems I: Regular Papers,* vol. 54, no. 1, pp. 48-59, 2007.

[62] K. T. Ng, F. Boussaid and A. & Bermak, "A CMOS single-chip gas recognition circuit for metal oxide gas sensor arrays," *IEEE Transactions on Circuits and Systems I: Regular Papers,* vol. 58, no. 7, pp. 1569-1580, 2011.

[63] A. Chortos, J. Liu and Z. & Bao, "Pursuing prosthetic electronic skin," *Nature materials,* vol. 15, no. 9, pp. 937-950, 2016.

[64] T. Kanao, H. Suto, K. Mizushima, H. Goto, T. Tanamoto and T. & Nagasawa, "Reservoir computing on spin-torque oscillator array," *Physical Review Applied,* vol. 12, no. 2, 2019.

[65] J. J. Blech, "On isothermal squeeze films," *Journal of lubrication technology,* vol. 105, no. 4, pp. 615-620, 1983.

[66] T. Veijola, H. Kuisma, J. Lahdenperä and T. & Ryhänen, "Equivalent-circuit model of the squeezed gas film in a silicon accelerometer," *Sensors and Actuators A: Physical,* vol. 48, no. 3, pp. 239-248, 1995.

[67] H. M. Ouakad and M. I. & Younis, "The dynamic behavior of MEMS arch resonators actuated electrically," *International Journal of Non-Linear Mechanics,* vol. 45, no. 7, pp. 704-713, 2010.

[68] M. I. Younis, MEMS linear and nonlinear statics and dynamics, Springer Science & Business Media, 2011.

[69] K. L. Turner, S. A. Miller, P. G. Hartwell, N. C. MacDonald, S. H. Strogatz and S. G. & Adams, " Five parametric resonances in a microelectromechanical system," *Nature,* vol. 396, no. 6707, pp. 149-152, 1998.

[70] M. I. Younis and F. & Alsaleem, "Exploration of new concepts for mass detection in electrostatically-actuated structures based on nonlinear phenomena," *Journal of computational and nonlinear dynamics,* vol. 4, no. 2, 2009.

[71] M. H. Hasan, F. M. Alsaleem and H. M. & Ouakad, "Novel threshold pressure sensors based on nonlinear dynamics of MEMS resonators," *Journal of Micromechanics and Microengineering,* vol. 28, no. 6, 2018.

[72] A. Ramini, M. I. Younis and Q. T. & Su, "A low-g electrostatically actuated resonant switch," *Smart Materials and Structures,* vol. 22, no. 2, 2012.

[73] F. M. Alsaleem, M. I. Younis and H. M. & Ouakad, "On the nonlinear resonances and dynamic pull-in of electrostatically actuated resonators," *Journal of Micromechanics and Microengineering,* vol. 19, no. 4, 2009.

[74] L. Medina, R. Gilat, B. Robert Ilic and S. & Krylov, "Experimental dynamic trapping of electrostatically actuated bistable micro-beams," *Applied physics letters,* vol. 108, no. 7, 2016.

[75] N. Bertschinger and T. & Natschläger, " Real-time computation at the edge of chaos in recurrent neural networks," *Neural computation,* vol. 16, no. 7, pp. 1413-1436, 2004.

[76] K. T. Alligood, J. A. Yorke and T. D. & Sauer, Chaos: An Introduction to Dynamical Systems, Springer, 2000.

[77] S. Achanta and S. V. & Gangashetty, " Deep Elman recurrent neural networks for statistical parametric speech synthesis," *Speech Communication,* vol. 93, pp. 31-42, 2017.

[78] R. D. Beer, "On the dynamics of small continuous-time recurrent neural networks," *Adaptive Behavior,* vol. 3, no. 4, pp. 469-509, 1995.

[79] H. Arie, J. Namikawa, T. Ogata, J. Tani and S. & Sugano, "Reinforcement learning algorithm with CTRNN in continuous action space," in *International Conference on Neural Information Processing*, Berlin, Heidelberg, 2006.

[80] A. E. Hoerl and R. W. & Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics,* vol. 12, no. 1, pp. 55-67, 1970.

[81] K. I. Funahashi and Y. & Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural networks,* vol. 6, no. 6, pp. 801-806, 1993.

[82] J. Boedecker, O. Obst, J. T. Lizier, N. M. Mayer and M. & Asada, "Information processing in echo state networks at the edge of chaos," *Theory in Biosciences,* vol. 131, no. 3, pp. 205-213, 2012.

[83] F. M. Alsaleem and M. I. & Younis, "Stabilization of electrostatic MEMS resonators using a delayed feedback controller," *Smart Materials and Structures,* vol. 19, no. 3, 2010.

[84] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre and P. & Bienstman, " Experimental demonstration of reservoir computing on a silicon photonics chip," *Nature communications,* vol. 5, no. 1, pp. 1-6, 2014.

[85] R. Ramesham and R. & Ghaffarian, "Challenges in interconnection and packaging of microelectromechanical systems (MEMS)," in *50th Electronic Components and Technology Conference*, 2000.

[86] R. D. Beer, "Toward the evolution of dynamical neural networks for minimally cognitive behavior," *From animals to animats,* vol. 4, pp. 421-429, 1996.

[87] R. D. Beer, "The dynamics of active categorical perception in an evolved model agent," *Adaptive behavior,* vol. 11, no. 4, pp. 209-243, 2003.

[88] K. Vandoorne, W. Dierckx, B. Schrauwen, D. Verstraeten, R. Baets, P. Bienstman and J. & Van Campenhout, "Toward optical signal processing using photonic reservoir computing," *Optics express,* vol. 16, no. 15, pp. 11182-11192, 2008.

[89] K. Hicke, M. A. Escalona-Morán, D. Brunner, M. C. Soriano, I. Fischer and C. R. & Mirasso, "Information processing using transient dynamics of semiconductor lasers subject to delayed feedback," *IEEE Journal of Selected Topics in Quantum Electronics,* vol. 19, no. 4, pp. 1501610-1501610, 2013.

[90] A. F. Atiya and A. G. & Parlos, " New results on recurrent network training: unifying the algorithms and accelerating convergence," *IEEE transactions on neural networks,* vol. 11, no. 3, pp. 697-709, 2000.

[91] H. Conrad, H. Schenk, B. Kaiser, S. Langa, M. Gaudet, Schimmanz, S. K. and M. M. & Lenz, "A small-gap electrostatic micro-actuator for large deflections," *Nature communications,* vol. 6, no. 1, pp. 1-7, 2015.

[92] D. Peroulis, S. P. Pacheco, K. Sarabandi and L. P. & Katehi, "Electromechanical considerations in developing low-voltage RF MEMS switches," *IEEE Transactions on microwave theory and techniques,* vol. 51, no. 1, pp. 259-270, 2003.

[93] J. M. Huang, K. M. Liew, C. H. Wong, S. Rajendran, M. J. Tan and A. Q. & Liu, "Mechanical design and optimization of capacitive micromachined switch," *Sensors and Actuators A: Physical,* vol. 93, no. 3, pp. 273-285, 2001.

[94] H. Hosaka, K. Itao and S. & Kuroda, "Damping characteristics of beam-shaped micro-oscillators," *Sensors and Actuators A: Physical,* vol. 49, no. 1-2, pp. 87-95, 1995.

[95] W. M. Van Spengen, R. Puers and I. & De Wolf, "A physical model to predict stiction in MEMS," *Journal of micromechanics and microengineering,* vol. 12, no. 5, p. 702, 2002.

[96] A. Eichler, J. Chaste, J. Moser and A. & Bachtold, "Parametric amplification and self-oscillation in a nanotube mechanical resonator," *Nano letters,* vol. 11, no. 7, pp. 2699-2703, 2011.

[97] R. B. Karabalin, X. L. Feng and M. L. & Roukes, "Parametric nanomechanical amplification at very high frequency," *Nano letters,* vol. 9, no. 9, pp. 3116-3123, 2009.

[98] A. H. Ramini, A. Z. Hajjaj and M. I. & Younis, "Tunable resonators for nonlinear modal interactions," *Scientific reports,* vol. 6, p. 34717, 2016.

[99] B. J. Gallacher, J. S. Burdess and K. M. & Harish, " A control scheme for a MEMS electrostatic resonant gyroscope excited using combined parametric excitation and harmonic forcing," *Journal of Micromechanics and Microengineering,* vol. 16, no. 2, p. 320, 2006.

[100] H. Liu, Y. Qian and C. & Lee, "A multi-frequency vibration-based MEMS electromagnetic energy harvesting device," *Sensors and Actuators A: Physical,* vol. 204, pp. 37-43, 2013.

[101] D. Marioli, E. Sardini, M. Serpelloni, B. Andò, S. Baglio, N. Savalli and C. & Trigona, "Hybrid telemetric MEMS for high temperature measurements into harsh industrial environments," in *2009 IEEE Instrumentation and Measurement Technology Conference*, 2009.

[102] S. Park and E. & Abdel-Rahman, "Low voltage electrostatic actuation and displacement measurement through resonant drive circuit," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2012.

[103] S. Park, M. Pallapa, J. T. Yeow and E. & Abdel-Rahman, "Low voltage electrostatic actuation and angular displacement measurement of micromirror coupled with resonant drive circuit," in *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*, 2012.

[104] N. Jaber, A. Ramini and M. I. & Younis, "Multifrequency excitation of a clamped–clamped microbeam: Analytical and experimental investigation," *Microsystems & nanoengineering,* vol. 2, no. 1, pp. 1-6, 2016.

[105] H. Jaeger, *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network,* Bonn: GMD-Forschungszentrum Informationstechnik: Approach(Vol. 5, p. 01), 2002.

[106] V. C. Müller and M. & Hoffmann, "What is morphological computation? On how the body contributes to cognition and control," *Artificial life,* vol. 23, no. 1, pp. 1-24, 2017.