**Proceedings of the ASME 2020**
**International Design Engineering Technical Conferences**
**and Computers and Information in Engineering Conference**
**IDETC/CIE2020**
**August 17-19, 2020, Virtual, Online**

**DETC2020-22665**

# MACHINE LEARNING AUGMENTATION IN MICRO-SENSOR ASSEMBLIES

**Mohammad H Hasan[1], Fadi Alsaleem[1], Amin Abbasalipour[2], Siavash Pourkamali Anaraki[2], Muhammad Emad-Un-Din[3], Roozbeh Jafari[3]**

[1]University of Nebraska – Lincoln, Omaha, NE, USA
[2]University of Texas – Dallas, Dallas, TX, USA
[3]Texas A&M University, College Station, TX, USA

## ABSTRACT

The size and power limitations in small electronic systems such as wearable devices limit their potential. Significant energy is lost utilizing current computational schemes in processes such as analog-to-digital conversion and wireless communication for cloud computing. Edge computing, where information is processed near the data sources, was shown to significantly enhance the performance of computational systems and reduce their power consumption. In this work, we push computation directly into the sensory node by presenting the use of an array of electrostatic Microelectromechanical systems (MEMS) sensors to perform colocalized sensing-and-computing. The MEMS network is operated around the pull-in regime to access the instability jump and the hysteresis available in this regime. Within this regime, the MEMS network is capable of emulating the response of the continuous-time recurrent neural network (CTRNN) computational scheme. The network is shown to be successful at classifying a quasi-static input acceleration waveform into square or triangle signals in the absence of digital processors. Our results show that the MEMS may be a viable solution for edge computing implementation without the need for digital electronics or micro-processors. Moreover, our results can be used as a basis for the development of new types of specialized MEMS sensors (ex: gesture recognition sensors)

## INTRODUCTION

The miniaturization of transistors and sensors has enabled the development of extremely compact wearable devices. These devices offer great potential to improve the quality of life of humans by monitoring operator health and performing diagnostics [1]. The potential of wearable devices is nonetheless hampered by the size and power limitations in commercial and research designs. Wearable devices are expected to perform complex classification tasks in real time while operating at extremely low power as to seldom require recharging or rely completely on energy harvesting as a power source.

Computational power limitation is often tackled by out-sourcing computation via clouding computing. This approach, while successful, is extremely power consuming [2,3] and may pause to some security concerns [4]. Recent works address these problems by focusing on optimizing the hardware of wearable devices by using energy-efficient components [5] or by optimizing the sampling rate of sensor data and the wireless communication rate with external devices [6]. These approaches serve to alleviate energy consumption problems. However, they do not address the energy inefficiency introduced in the analog-to-digital conversion (ADC) and wireless communication processes. Moreover, lowering the rates of data sampling reduces the accuracy of algorithms that require fast processing rate such as heart rate monitoring and fall detection.

Therefore, to reduce the need for the computationally expensive ADC and digital processing, there is a great need for analog sensors capable of performing computation on the edge and can easily be interfaced with digital computing devices. Microelectromechanical systems (MEMS) devices are prime candidates for utilization in this scheme as they are already used as sensory elements in wearable devices and smart systems. Moreover, individual MEMS devices were shown to have computationally attractive features that resemble those of continuous-time recurrent neurons (CTRNs) [7,8].

This work tackles the problem of performing energy-efficient computation by relying on the highly energy-efficient smart MEMS networks to perform high-level computational tasks in situ, and then sending the processing units the pre-processed unit at a significantly lower rate. This work is inspired by the use of MEMS sensors as threshold switches [9] and digital accelerometers [10], which may be viewed as early examples of computation at the sensor level. However, in this work, more intensive computing is required.

The organization of this paper is as follows: first, we introduce the formulation of the MEMS computing scheme by providing a computational model of coupled MEMS dynamics. Next, we optimize a small network of MEMS devices to classify an acceleration waveform into square and triangle signals. We then show the classification results using the MEMS network. Finally, we discuss our results and conclude the paper.

## NETWORK FORMULATION

The dynamics of a single electrostatic some MEMS device can be modeled as a single degree-of-freedom spring-mass-damper system, as shown in Figure 1. When placed in a network of $N$ MEMS devices, the equation of motion of the $i^{th}$ MEMS device is given by (1):

$$m_i\ddot{x}_i(t) + c_i\dot{x}_i(t) + k_i x_i(t) = \frac{\varepsilon A_i(V_i(t))^2}{2\left(d_i - x_i(t)\right)^2} - m_i\ddot{y}(t) \quad i = 1,2,\dots,N \qquad (1)$$

where $m_i$, $c_i$ and $k_i$ are the effective mass, damping constant and linear stiffness of the $i^{\text{th}}$ MEMS device in the assembly of $N$ MEMS devices, respectively. $x_i(t)$ is the deflection the MEMS device at time $t$, $\varepsilon$ is the electrical permittivity, $A_i$ is the overlapping electrode area, $d_i$ is nominal separation distance between MEMS electrodes, $V_i(t)$ is the effective voltage acting on MEMS $i$, and $\ddot{y}(t)$ is the base acceleration.

The electrostatic forcing in (1) results in a singularity when the two MEMS electrodes come in contact, or when $x_i(t) = d_i$, named the pull-in instability. This instability, represented by a sudden response jump, has been shown to be useful when operating a MEMS sensor as a threshold switch. The pull-in regime is also known to have hysteresis; the pull-in voltage is higher than the release voltage. Switching instability and memory retention via hysteresis has been shown two of the most important properties of a class of artificial neurons named continuous-time recurrent neurons (CTRNs), which form the building block of a non-conventional computing scheme named continuous-time recurrent neural networks (CTRNNs) [7.8]. As these features inherently exist within the pull-in regime, this work focuses on operating a MEMS network around the pull-in regimes.

To eliminate the pull-in singularity in simulation and avoid electrical contact in practice, stoppers are installed in each MEMS device at a distance $x_{s,i}$. As such, (1) is overridden to $x_i(t) = x_{s_i}$ and $\dot{x}_i(t) = 0$ if it was found that $x_i(t) > x_{s,i}$.

Coupling MEMS devices is essential to emulate the behavior of a CTRNN and to complete the network. Here, MEMS devices are coupled electrically via the term $V_i(t)$ as shown in (2):

$$V_i(t) = V_{bias,i} + \sum_{j=1,j\neq i}^{N} w_{ij} V_{out,j}(t) \qquad (2)$$

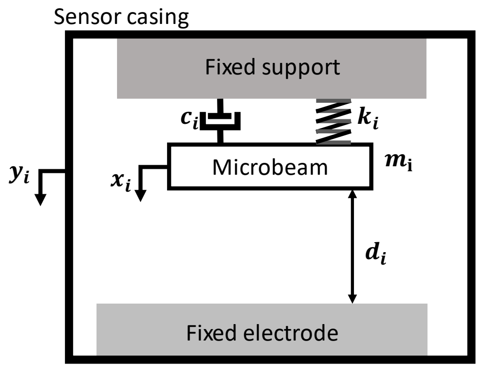



Figure 1. MEMS model as a single degree-of-freedom spring-mass-damper system.

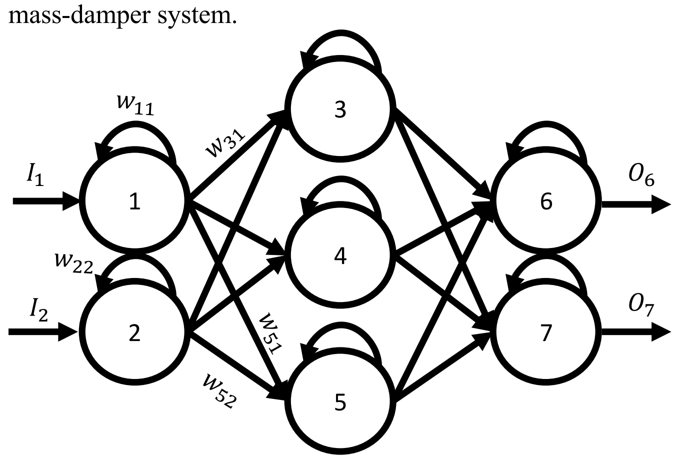



Figure 2. A MEMS network example. Each numbered node represents a MEMS device. Here, $\mathrm{i} = 1,2,\dots,7$. The figure contains some connection labels. The network contains two inputs, $\mathrm{I}_1$ and $\mathrm{I}_2$ and 2 outputs $\mathrm{O}_6$ and $\mathrm{O}_7$.

where $V_{bias,i}$ 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 (3):

$$V_{out,j}(t) = V_{bias_j} U\left(x_j(t) - x_{s,j}\right) \qquad (3)$$

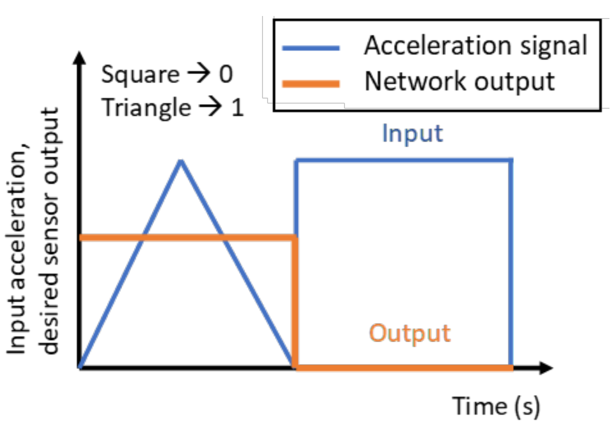
where $U(.)$ is a unit step function. We note here that self-connection, typically given by $w_{ii}$, is essential for computation. While implicit, this recurrent connection is observed in the pull-in regime as evidenced by hysteresis. Here, the MEMS connections are forward and unidirectional (aside from the implicit self-feedback connection). Therefore, $w_{ij} = 0$ if $j > 0$, forming layers in the network, much like simulated CTRNNs, as shown in Figure 2.

We note here 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.

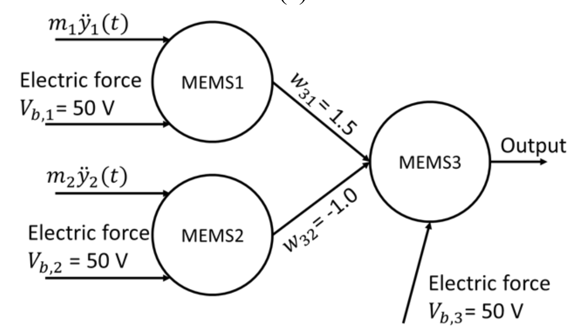
## COMPUTATIONAL TASK AND NETWORK STRUCTURE

Classification is one of the most popular tasks in the machine learning literature. For this work, we consider a simple classification task as a test for computational potential of a network of MEMS devices. The task here involves classifying an input waveform into either 'Square' signal or 'Triangle' signal, as shown in Figure 3. The input waveforms are supplied as acceleration waveforms. We note here that, 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.

The bias voltages were chosen such that $V_{b,1} > V_{b,2}$ 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 [12,13]. 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.




(a)




(b)

Figure 3. 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.

## CLASSIFICATION TASK USING A MEMS CTRNN

For our task, a network of identical MEMS accelerometer devices was used. The parameters of the MEMS devices are presented in table 1. Additional information about the sensors used can be found in [14]. 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 $\ddot{y} = -5g$. The results of the MEMS CTRNN are shown in Figure 4. The shock signal excites both MEMS1 and MEMS2 (Figure 4,a and 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 $V_3(t)$. 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 $w_{31}V_{b,1} + w_{3,2}V_{b,2} + V_{b,3}$. 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

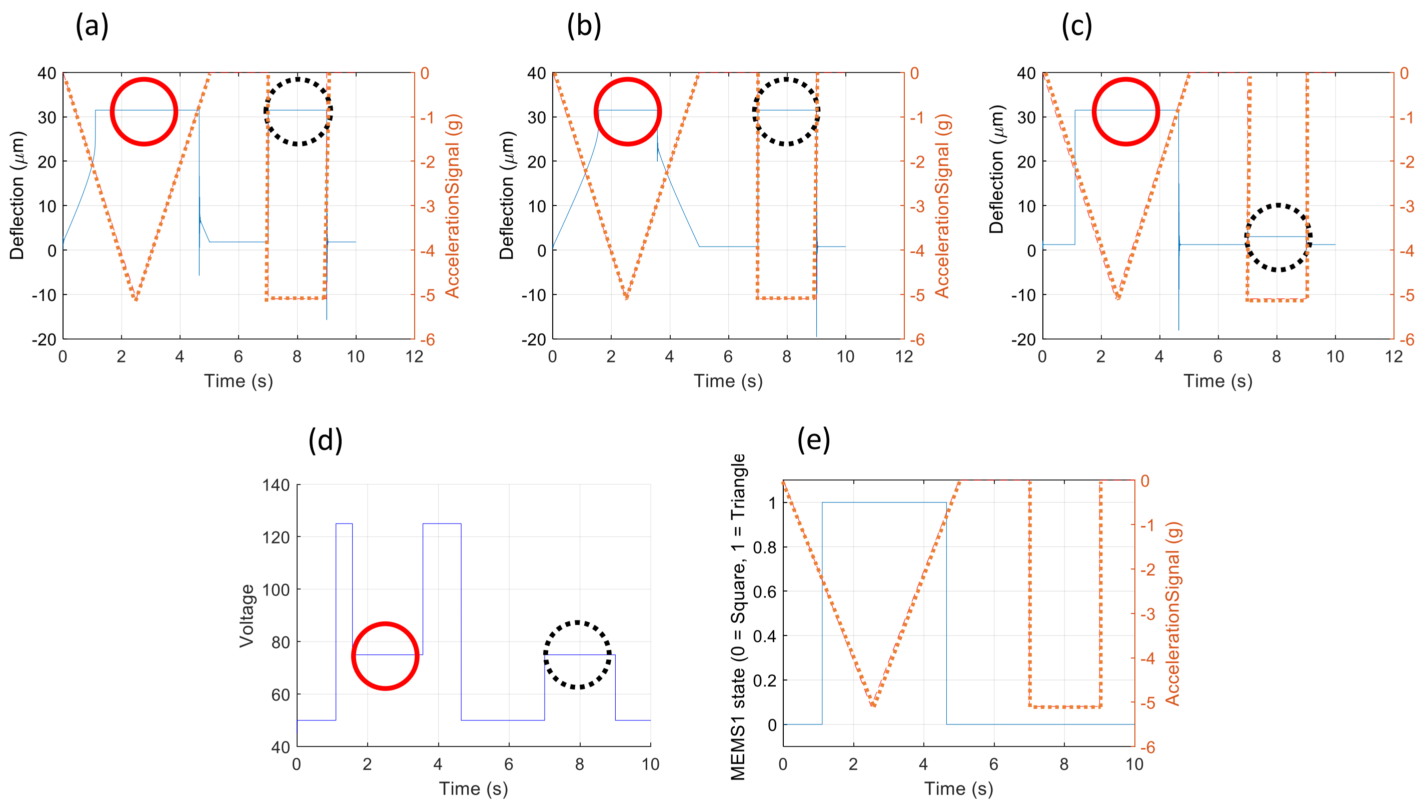



Figure 4. Classification test results showing the response of MEMS1 (a), MEMS2 (b) and MEMS3 (c). (d) The effective votlage acting on MEMS3 $V_3(t)$. (e) The state of MEMS3 when subject to a triangle or a square signal. Note: the points marked by red and black circles in (a-d) represent points with similar MEMS1 and MEMS2 states but different MEMS3 states, indicating the importance of memory in a MEMS CTRNN.

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 shakers which tend to be trapezoidal in shape, assuming the signal ramp is sufficiently steep, since the MEMS devices will slightly lag the input signal.

Table 1: MEMS parameters.

| **MEMS Parameter** | **Value** |
|---|---|
| $l$ | 9 mm |
| $b$ | 5.32 mm |
| $\varepsilon$ | $8.85\times10^{-12}$ F/m |
| $d$ | 42 $\mu$m |
| $k$ | 215 N/m |
| $m$ | 143 mg |
| $c$ | 0.351 N.s/m |
| $V_{b,1}$ | 50 V |
| $V_{b,2}$ | 50 V |
| $V_{b,3}$ | 50 V |
| $w_{31}$ | 1.5 |
| $w_{32}$ | -1 |
| $x_s$ | 30 $\mu m$ |

The results from Figure 4 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 Figure 4a-d, in which MEMS1 and MEMS2 are simultaneously pulled-in, yet MEMS3 can assume two different configurations).

## DISCUSSION AND CONCLUSION

This work presents a new class of MEMS sensory arrays capable of performing non-trivial computational tasks at the sensor level. The designed sensory array exploits 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). Coupling MEMS within an array enables non-conventional computing using the MEMS dynamics, in an analog fashion, thus eliminating the need for some analog-to-digital conversion.

For simple tasks, training such a binary MEMS network is simple using ladder logic as a starting point. Additional modifications by considering MEMS dynamics can reduce the size of the network. The computational task considered in this work involves a simple binary classification of quasi-static square and triangle acceleration signals. We show that the trained MEMS network is capable of classifying the input signal even in regimes in which the states of the input-layer MEMS devices (MEMS1 and MEMS2) are identical due to memory retention at the pull-in regime.

This work 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).

## REFERENCES

[1] Green, E. M., van Mourik, R., Wolfus, C., Heitner, S. B., Dur, O., & Semigran, M. J., 2017, "Machine learning detection of obstructive hypertrophic cardiomyopathy using a wearable biosensor. Circulation", 136(suppl_1), pp. A24031-A24031.

[2] Shi, W., & Dustdar, S., 2016, "The promise of edge computing. Computer", 49(5), pp. 78-81.

[3] Kalantarian, H., Sideris, C., Mortazavi, B., Alshurafa, N., & Sarrafzadeh, M., 2016, "Dynamic computation offloading for low-power wearable health monitoring systems". IEEE Transactions on Biomedical Engineering, 64(3), pp. 621-628.

[4] Zhou, J., Cao, Z., Dong, X., & Lin, X., 2015, "Security and privacy in cloud-assisted wireless wearable communications: Challenges, solutions, and future directions", IEEE wireless Communications, 22(2), pp. 136-144.

[5] Maurer, U., Rowe, A., Smailagic, A., & Siewiorek, D. P. ,2006, "eWatch: a wearable sensor and notification platform", International Workshop on Wearable and Implantable Body Sensor Networks, April 3, 2006, (BSN'06), IEEE, pp. 4-pp.

[6] Fafoutis, X., Marchegiani, L., Elsts, A., Pope, J., Piechocki, R., & Craddock, I. ,2018, "Extending the battery lifetime of wearable sensors with embedded machine learning". 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), February, 2018, IEEE, pp. 269-274.

[7] Alsaleem, F., Hasan, M. H., Tesfay, M., & Rafaie, M. , 2018, "MEMS As a Continuous Time Recurrent Neuron (CTRN) Computing Unit." Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Quebec City, Quebec, Canada. August 26–29, 2018. ASME paper no. DETC2018-85075.

[8] Alsaleem, F. M., Hasan, M. H., & Tesfay, M. K. ,2018, "A MEMS nonlinear dynamic approach for neural computing". Journal of Microelectromechanical Systems, 27(5), pp. 780-789.

[9] Younis, M. I., Alsaleem, F. M., Miles, R., & Su, Q. ,2007, "Characterization of the performance of capacitive switches activated by mechanical shock". Journal of Micromechanics and Microengineering, 17(7), pp. 1360.

[10] Kumar, V., Jafari, R., & Pourkamali, S. ,2016, "Ultra-low power digitally operated tunable MEMS accelerometer". IEEE Sensors Journal, 16(24), pp. 8715-8721.

[12] G. Schöner, J. Spencer, and A. Schutte, 2015, "Dynamic Thinking", 1st ed. London, U.K.: Oxford Univ. Press.

[13] G. Schöner, 2008, "Dynamical systems approaches to cognition," in Toward a New Unified Theory of Development: Connectionism and Dynamical Systems Theory Re-Considered. New York, NY, USA: Oxford Univ. Press.

[14] Alsaleem, F. M., Younis, M. I., & Ouakad, H. M., 2009, "On the nonlinear resonances and dynamic pull-in of electrostatically actuated resonators". Journal of Micromechanics and Microengineering, 19(4), pp. 045013.

