Removing Unclassified Hand Tremor Motion from Computer Mouse Input with Neural Networks

Stephen Mack*

Electrical Engineer, Lawrenceville NJ (USA)

Received 17 May 2018 | Accepted 3 October 2018 | Published 19 October 2018

Abstract

An artificial neural network based filter to remove unwanted tremor-induced motion in computer mouse input is presented and tested. A method to efficiently capture appropriate training data is shown to be important in the operation and training of the neural network filter. The architecture of the neural network as well as the numerous design choices are presented and explained. A simulation study proves the artificial neural network is successful at removing a simulated Parkinson's tremor from computer mouse movements even with minimal training data. Resulting tremor-free motion estimated by the artificial neural network is shown to be similar to normal tremor free computer mouse movements.

Keywords

Filter, Human Computer Interface, Neural Network, Tremor, Mouse Cursor.

DOI: 10.9781/ijimai.2018.10.003

I. Introduction

USING a computer mouse accurately and quickly is a vital part of modern day life that many people take for granted. Conditions that affect muscular control or cause hand tremors can hinder a person's ability to use a standard computer mouse, limiting their ability to work or participate in aspects of modern life. Many government processes and forms are now only available online, making interfacing with a computer a daily necessity. It is estimated that 7 million Americans, or roughly 2.2% of the American population, has some form of an essential tremor (ET) [1]. These 7 million Americans have a wide range of condition symptoms and severities, which makes it difficult to provide universal assistance solutions.

Some people with tremors have found that using a track-ball style mouse works better than a standard mouse, but this costs money and the person still needs to learn how to effectively use it. Other researchers believe that an adjustable user interface can solve the problem [2], but while this is a solution it is not allowing the user to resume normal use of their computer. The optimal solution will be to allow the person to continue to use a standard computer mouse that they are used to, and remove the tremor-induced motion in software.

Much work has been done to determine how different conditions can affect the use of a computer mouse [3][4][5][6]. This research has improved our statistical understanding of how these tremors affect computer mouse movements and provide insight into the range of tremor severities for different conditions. Such statistical understanding of how these tremors affect computer mouse inputs has led to incredible advancements in diagnosing these conditions by simply analyzing a user's mouse movements [7][8], some have even attempted to diagnose these conditions by analyzing the users physical movements [9]. Diagnostic developments like these have the potential to lower the cost and complexity while increasing the speed of testing patients.

* Corresponding author.

E-mail address: SteveCRM@gmail.com

Rapid and accurate diagnostic achievements such as these are a large step forward in the field, but to date very little has been done to actively remove the unwanted tremor motion. One of the main problems is that there is such a wide range of types of tremors, all with varying levels of intensity and frequencies making it hard to account for them all. For example, Parkinson's tremors are known to exist from 3-7Hz, while orthostatic tremors range from 13-18Hz [10]. Characteristics of the tremor will also vary person to person depending on severity of the condition.

Traditional filtering techniques such as moving average filters, infinite impulse response (IIR) and finite impulse response (FIR) filters can easily attempt to remove the unwanted frequencies, but these filter designs are not easily adjustable or customizable to a specific patient's tremor and usually produce sub-optimal results. Some software packages that claim to be able to remove tremor-induced motion from a computer mouse exist, but all have a set number of filters for the user to choose from [11]. While these software packages may work well, they will never be truly tailored to an individual user, and a person with a tremor can only hope that one of these predetermined filters will fit their tremor well enough.

A truly universal software solution would need to accurately model a user's tremor, recognize the patterns in the tremor to remove them, and function smoothly and continuously in time. Artificial neural networks have been shown to be effective at prediction, modeling, pattern matching tasks [12], and have been useful in time-series tasks [13][14][15], making them a good choice for removing tremor-induced motion from standard computer mouse inputs.

This paper describes the design of a multilayer artificial neural network that is capable of being trained by the backpropagation algorithm to remove the unwanted mouse cursor movements caused by essential tremors in an individual patient or a generalized group of patients. A simulation study is offered that demonstrates the capability of removing the unwanted mouse cursor tremor from an individual user with a simulated Parkinson's tremor. Results of the simulation study demonstrate the power and potential uses of this filtering technique.

The following sections present and verify this new method of removing unwanted tremor induced motion from a computer mouse based on artificial neural networks that can be customized for an individual patient.

II. Problem Statement

This paper presents an artificial neural network and corresponding training method to remove tremor-induced noise from the motion of a standard computer mouse. Section III presents the architecture of the neural network. Section IV describes how the neural network will be trained, and provide methods for collecting suitable training data. Section V details a simulation study to prove the effectiveness of the presented neural network solution. Finally, Section VI presents conclusions and avenues for future research.

III. Design of Neural Network

Artificial neural networks are an effective method at modeling nonlinear mathematical functions. The neural network presented here will attempt to accurately model the nonlinear relationship between computer mouse input with and without a tremor.

The output of the neural network will be an estimate of tremor-free mouse cursor motion, and it will make this estimate with a recent history of cursor motion with tremor. The artificial neural network takes in recent time samples in parallel, meaning each input neuron will receive a recent historical mouse sample. How many historical samples used as inputs can be varied to achieve the best results, but it is recommended that it should be at least a second or more to capture enough information about tremors of a wide range of frequencies and the desired motion of the mouse.

Because the neural network will be analyzing the motion of the cursor it is important that the historical samples used as inputs be cursor velocities instead of cursor positions. Velocity is a better input choice because we care about where the cursor is going and how quickly, and as the cursor rapidly changes direction the input will be normalized around 0 pixels per second. If position were to be used we would need training data for all areas of the screen evenly, and would need to retrain the entire network if the user changed screen sizes or resolutions. This amount of training data is technically feasible, but not realistic for the goal of making a simple and easy to configure universal filter to remove tremors from the computer mouse movements.

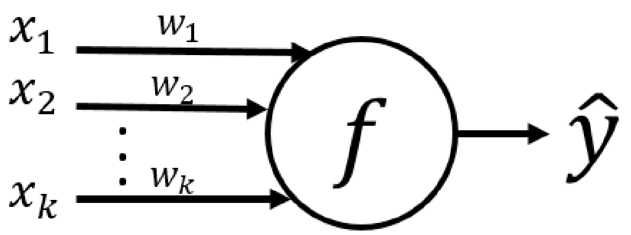


Fig. 1. Visual model of an artificial neuron.

The artificial neural network is made up of multiple neurons, usually in multiple layers. The output of an artificial neuron is defined by Eq. (1), and Fig. 1 provides a graphical.

$$\hat{y} = f(w^T x) \tag{1}$$

where f is the activation function, x is the input matrix, $\hat{y}$ is the output, and w is the matrix of weights associated with the inputs. The error of the neuron is given in Eq. (2).

$$e = y - \hat{y} \tag{2}$$

where y is the known training data.

Training the neural network means that all of the weights will have to be carefully chosen to approximate the appropriate nonlinear function to remove the tremor from the computer mouse input motion. The backpropagation algorithm is used to train artificial neural networks, using the gradient descent algorithm to determine and update the neural network weights. The weights in the artificial neural network are updated using Eq. (3).

$$w_k^{new} = w_k^{old} - n\frac{de}{dw} \tag{3}$$

where n is the step size, e is neuron error, and k is the neuron index.

Choice of activation function can greatly affect the performance of a neural network. If all neurons use a linear activation function then the entire network will only be able to approximate a linear function, no matter how large the network is. This means that it is vitally important that the neural network use nonlinear activation functions for the majority of the network. To ensure that the network is capable of approximating a nonlinear relationship the rest of the neurons in the network are given a nonlinear activation function, in this case the rectified linear function (ReLu) as shown in Eq. (4). The output of the neural network is an estimate of tremor-free cursor velocity, and because the velocity will theoretically have no upper or lower limit the single output neuron will use a linear activation function.

$$f(x) = \max(0, x) \tag{4}$$

If this neural network is to be designed to fit a single user then it is likely that there will be a realistic limit to the amount of training data available, a user will not sit for hours to provide training data. With small amounts of training data there is a very real possibility that the neural network can suffer from over-fitting where it will predict specific behaviors in the training data rather than the general trend in the training data. To combat over-fitting of the neural network and to ensure that the network output is not too sensitive to a small subset of neuron weights, a 20% dropout is used between each layer of the neural network.

The total number of neurons in the network will depend on the number of layers, and the number of neurons in each layer. There will be an input layer that will have as many neurons as there are historical input samples, and an output layer with a single neuron. All the layers in-between are referred to as hidden layers. The number of hidden layers and the number of neurons in each layer will affect how well the overall neural network can remove the tremor motion, but it can also affect the amount of computation time needed to make a prediction. If the neural network is to work in real time there are definite time constraints to creating a prediction. Future research and testing will be needed to optimize these values for a functional software prototype.

The neural network is estimating the velocity of the mouse cursor, meaning that to find the most recent cursor position we need to sum the velocity with the last known cursor position as shown in Eq. (5).

$$x[k] = x[k-1] + \hat{y}[k] \tag{5}$$

where x is the cursor position, k is the current sample, and $\hat{y}$ is the estimated cursor velocity. It is important to note that the cursor position is never updated with a true cursor position from the operating system, but only from an initial starting point and velocity estimates from the neural network. Similar to many dead reckoning algorithms, updating in this way means that any error in the neural network estimate will create a constantly growing error drift in cursor position. This drift is likely to be small enough that the user will be able to naturally and unknowingly account for it with low frequency adjustments. This issue will need to be thoroughly tested in future research.

IV. Data Collection & Neural Network Training

The presented artificial neural network is to be trained with the backpropagation algorithm, which means that input data as well as classified output data is required to model the tremor filtering behavior. Input data will simply be a user's recent mouse velocities with tremor motion, and the output data will be the correct next cursor velocity if the user had no tremor.

The input data is simple to collect, but it is much more difficult to obtain acceptable output data for training. Traditional filtering methods could be applied to the input in an attempt to recreate a cursor velocity without a tremor but then the neural network would be approximating known methods, providing little additional value to the state of the art. The best obtainable data to represent tremor free mouse motion is the user's desired cursor velocity. Desired location is not a simple concept to measure, unless a desired location is provided for the user to track with their cursor.

To collect mouse movements with tremors as well as the desired movement, a software package was developed to record a single user's mouse cursor and tracking target location data over a short period of time. The software asked the user to track a target on the screen as best they can, trying to keep their cursor as close to the cross-hair as possible with their unique tremor.

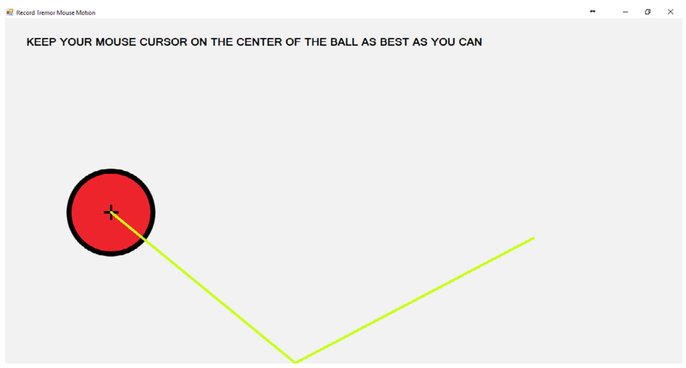
The target moves around the screen in regular straight lines, only changing directions when it reaches the end of the screen where it appears to bounce off at a predictable angle. The ball also slowly changes velocity to ensure that data is collected over all ranges of normal cursor speeds. Changes in target position and velocity are meant to be extremely predictable to the user so that any cursor pointing error derives from the hand tremor, and not uncertainty in the target motion. To sufficiently train a neural network to remove tremor motion and give the user normal cursor motion we must collect data covering all situations that are considered normal cursor motion. Changing direction and speed of the target ensures that the neural network will be trained over all normal mouse cursor movement situations.

To assist the user in accurately tracking the target the software removes any distractions by taking up the whole screen with a single color. The target is made easy to visually track by making it bright red with a bold black crosshair in the center. To assist the user in predicting the motion of the target a green tail shows the targets most recent motion. With these aids we attempt to assure that any error in pointing is from the patient's tremor, and not any other external factor.

Cursor data is collected for as long as the user would like, though the first and last ten seconds of data are removed so that any cursor position error coming from the user purposefully leaving the target to hit the start or close buttons is ignored. Beyond this trimming of the recorded data the software in no way alters the cursor data, it simply records the cursor location and the target location.

The resulting data set is a time stamped list of the location of the two dimensional mouse cursor location as well as the desired target location. Similar data sets are collected by researchers attempting to diagnose specific tremor conditions, but in this case the data will be used to compensate for the existing tremor. A screenshot of this software collecting data can be seen in Fig. 2.

This method can be used on a single patient to personally customize their software, or could be used for large scale data collection programs to create more universal neural network filters. Ideally the neural network will be trained with as much data as is pragmatic, as training with too little data can lead to poor performance.



KEEP YOUR MOUSE CURSOR ON THE CENTER OF THE BALL AS BEST AS YOU CAN

Fig. 2. Screenshot of software recording mouse data with moving target as desired location.

Collecting this data is not a computationally intensive process, though training a neural network can easily become an unreasonable task for a low-end home computer. The difference in computational abilities means that for a final product it would be beneficial to collect the training data on a user's machine, but offload the neural network training to a dedicated server off-site. For such a setup to work the training data would need to be collected at the user's computer, then uploaded to a server where the neural network would be trained, and finally the server would send back a trained neural network to the user when complete. Currently the software only records training data to a file, but can be turned into a service by sending the results over the internet to a waiting server with neural network training software.

V. Simulation Study

A simulation was created to demonstrate the ability of the artificial neural network to remove a simulated Parkinson's tremor from computer mouse movement. For simplification and presentation purposes this simulation study focuses on removing tremor only from the horizontal axis of the mouse cursor data, though it would not be difficult to adjust this study to accommodate the two axes of motion.

This study aims to remove a simulated a 6Hz Parkinson's tremor from a test patient's mouse cursor movements. Data used for this simulation was collected using the software described in section IV. The previously presented data collection software records cursor positions every 15.5ms, or a sample rate of 64.5Hz, and converts them to cursor velocities to be saved to a file. This sampling rate is sufficiently greater than the Nyquist sampling rate of the 6Hz Parkinson's tremor. Two sets of data must be collected, one to train the neural network and a second to validate the trained neural network.

The first data set will use the cursor velocities with tremor as well as the desired velocities to train the neural network. Once the network is trained it will use the second data set to validate its predictions of tremor free motion. It is important to use two different data sets for training and validation so that we can be sure that the neural network is learning the underlying dynamics of removing tremor motion and not simply memorizing and regurgitating the data we gave it during training like we would see in severely over-fit models.

Training data was collected for 32 seconds producing 2089 samples, and validation data was collected for 26 seconds producing 1661 samples.

A portion of the validation data set showing the cursor position error is shown in Fig. 3.

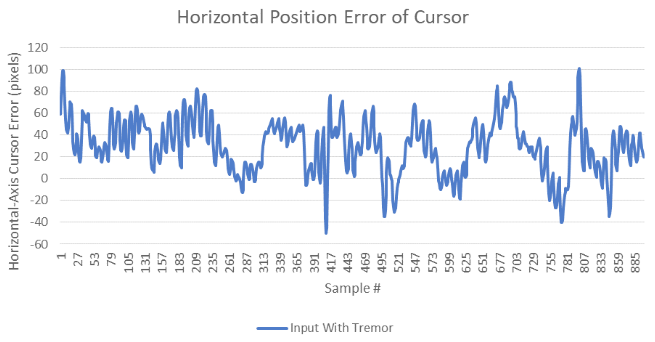


Fig. 3. Validation data set horizontal position error of cursor.

Cursor position error is centered near zero though there is obviously a high frequency component to the motion, unlike what would be seen in a user without a tremor. There are also long periods where the error is obviously purely positive or negative, meaning the user had difficulty tracking exactly on the target.

The artificial neural network in this simulation is comprised of 5 layers: 20 input nodes taking in the 20 most recent cursor velocity samples, 25 nodes in layer 2, 16 nodes in layer 3, 16 nodes in layer 4, and a single output node. A 20% dropout factor is added between each layer to prevent over-fitting. The output node uses a linear activation function, all other neurons use the rectified linear activation function (ReLu).

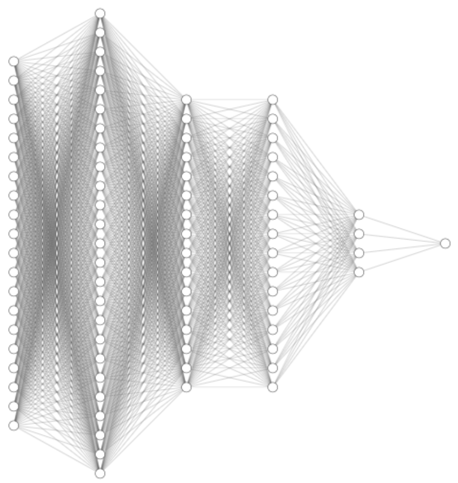


Fig. 4. Architecture of the simulated neural network.

Fig. 4 depicts a visual representation of the proposed neural network. Each circle represents a neuron, and each line represents an interconnection with an associated weight. The input layer is represented on the left, with the single output neuron on the right.

The artificial neural network is then trained with the training data set using python and keras software packages. Once the neural network was sufficiently trained, it was used to predict the tremor-free cursor velocities for the validation data set. For visualization purposes the velocities have been converted back into position errors measured in pixels. Fig. 5 shows the same horizontal cursor position error from the validation data set as in Fig. 3, but also includes the error of the validation data set after the neural network attempted to remove the tremor.

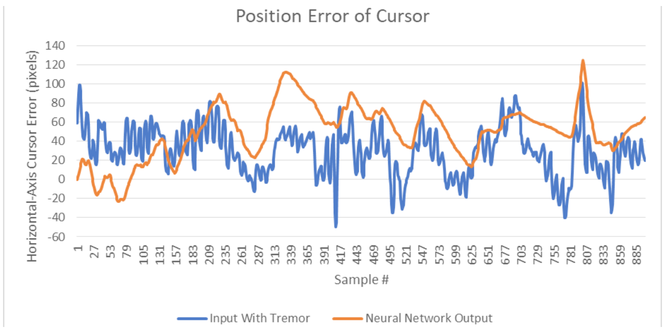


Fig. 5. Horizontal cursor position error before and after neural network improvement.

As shown in Fig. 5, the neural network successfully removed much of the higher frequency cursor motion, creating a much smoother looking position error graph. While the high frequency components of the error have been removed there does appear to be some offset error as predicted in section III. The offset error is relatively small and low frequency as expected, meaning the user could likely compensate for the effect without realizing.

While visually it appears that the higher frequency components of the cursor motion have been removed we still need to verify that the simulated 6Hz Parkinson's tremor has been sufficiently suppressed. To quantify the amount of tremor that was inhibited by the neural network a Fourier analysis is performed on the input with tremor as well as the neural network output without tremor. The Fourier analysis was done on using cursor position data rather than cursor velocity for visualization purposes. As shown in Fig. 6, the validation set has a tremor between 5Hz and 6Hz, consistent with tremors in patients with Parkinson's disease. The output of the trained neural network showed almost no motion at 6Hz, meaning the artificial neural network successfully filtered the tremor induced motion.

It is also interesting to note that the spectral intensity at 4 Hz and 8 Hz is roughly the same for each plot, meaning the neural network isn't simply acting as a low pass filter, but has learned to target the tremor specifically around 6 Hz.

Fig. 6 shows that the simulated Parkinson's tremor was removed but does not prove that the resulting mouse movement is similar to natural tremor-free computer mouse motions. To prove the neural network output is similar to natural mouse movements Fourier analysis is used to compare the output of the trained neural network to the desired cursor position. Fig. 7 demonstrates that the neural network produces similar frequency components to the tracking target motion, meaning the mouse should act like a regular mouse cursor to the user.

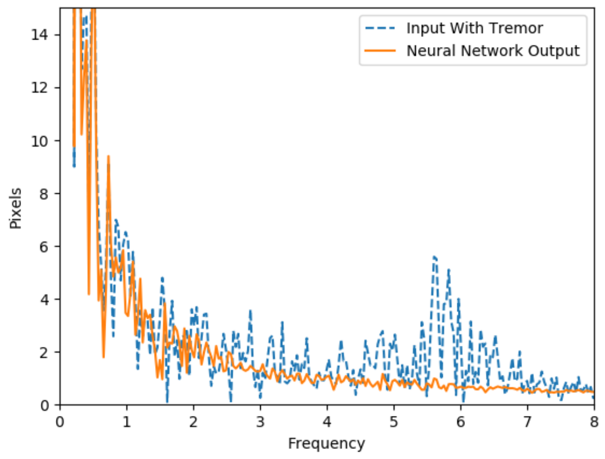


Fig. 6. Fourier analysis – removing tremor.

Key to note is that above 2 Hz the desired target motion and the neural network estimated motion are nearly identical. Traditional filtering methods would likely have acted as band-stop filters, suppressed the entire region where the tremor is present, but instead of simply removing spectral information the neural network actually returns the tremor motion back to a normal non-tremor state.

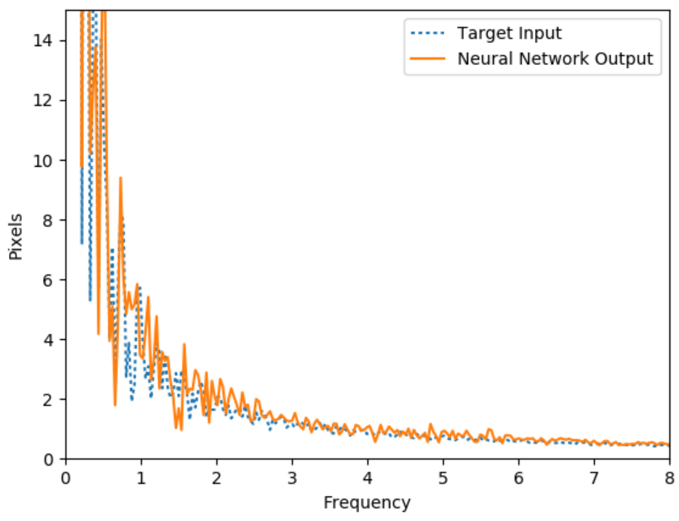


Fig. 7. Fourier analysis – performing like standard computer mouse.

The simulation study demonstrates that the proposed neural network design can effectively remove simulated Parkinson's tremor motion in one axis of a computer mouse cursor, and that the resulting tremor-free motion is similar to normal computer mouse movements.

VI. Conclusions & Future Work

The concept of using a neural network to remove tremors from computer mouse input has been shown to be extremely effective even with minimal training data from a single user. Not only did the neural network remove the tremor but the resulting mouse motion closely mimicked normal mouse movements, meaning a user will hopefully experience no discomfort or clumsiness while using a computer.

Additional research must be conducted to find an optimal size and structure of the neural network. A neural network larger than necessary will be a waste of the user's computer resources, and runs the risk of not being able to keep up with the speed at which cursor position samples are arriving. Alternative number of hidden layers, number of neurons per layer, and choice of activation function could also improve the functionality of the design.

In addition to the success of the neural network, the data collection software has also proven to be effective at collecting mouse movements with tremor while also supplying an accurate representation of the user's desired cursor motion. The software can be easily connected to the internet and scaled up for research into more universal filters, or for implementation into a consumer product.

The ability to design an easily customizable and situation specific filter without fully characterizing the noise opens this work up many possible applications beyond removing tremors from computer mice. New research avenues into neural network based noise cancellation can be applied to fields in signal processing, control theory, and many others.

Work on this topic will continue with the creation of a network capable of handling horizontal and vertical mouse movements, as well as creating a real time functioning prototype. With a functioning prototype experimentation on developing a more universal filter can be explored. A prototype will also allow patients with confirmed and medically diagnosed tremors to validate the concept and provide valuable feedback on functionality.

References

[1] Louis, Elan D., and Ruth Ottman. "How Many People in the USA Have Essential Tremor? Deriving a Population Estimate Based on Epidemiological Data." *Tremor and Other Hyperkinetic Movements*, 2014, Vol. 4, pp. 259.

[2] Aqueasha Martin-Hammond, Foad Hamidi, Tejas Bhalerao, Christian Ortega, Abdullah Ali, Catherine Hornback, Casey Means, and Amy Hurst, "Designing an Adaptive Web Navigation Interface for Users with Variable Pointing Performance". *Proceedings of the Internet of Accessible Things 2018*, pp. 31.

[3] Simeon Keates and Shari Trewin. "Effect of age and Parkinson's disease on cursor positioning using a mouse". *In Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility,* 2005, pp. 68-75.

[4] Alvaro D. Taveira, Sang D. Choi, "Review Study of Computer Input Devices and Older Users", *International Journal of Human–Computer Interaction*, 2009, Vol. 25, no. 5, pp. 455-474.

[5] R. C. Eberhart and Xiaohui Hu, "Human tremor analysis using particle swarm optimization," *Proceedings of the 1999 Congress on Evolutionary Computation,* 1999, Vol. 3, pp. 1927-1930.

[6] Cohen O, Pullman S, Jurewicz E, Watner D, Louis ED. "Rest Tremor in Patients With Essential Tremor" *Archives of Neurology*. 2003, Vol. 60, no. 3, pp. 405–410.

[7] N. M. Aly, J. R. Playfer, S. L. Smith, D. M. Halliday; "A novel computer-based technique for the assessment of tremor in Parkinson's disease", *Age and Ageing*, 2007, Vol. 36, no. 4, pp. 395–399.

[8] Hirschauer, T.J., Adeli, H. & Buford, J.A. "Computer-Aided Diagnosis of a Parkinson's Disease Using Enhanced Probabilistic Neural Network" *Journal of Medical Systems*, 2015, Vol. 39, no. 11, pp. 179.

[9] Enas Abdulhay, N. Arunkumar, Kumaravelu Narasimhan, Elamaran Vellaiappan, V. Venkatraman, "Gait and tremor investigation using machine learning techniques for the diagnosis of Parkinson disease", *Future Generation Computer Systems*, 2018, Vol. 83, pp. 366-373.

[10] Seyed Yashar Bani hashem MEng, Nor Azan Mat Zin PhD, Noor Faezah Mohd Yatim PhD & Norlinah Mohamed Ibrahim MD, "Improving Mouse Controlling and Movement for People with Parkinson's Disease and Involuntary Tremor Using Adaptive Path Smoothing Technique via B-Spline", *Assistive Technology: The Official Journal of RESNA*, 2014, Vol. 26, no. 2, pp. 96-104,

[11] Lee, Hong Ji et al. "Tremor frequency characteristics in Parkinson's disease under resting-state and stress-state conditions", *Journal of the Neurological Sciences*, 2016, pp. 272 - 277.

[12] B. Widrow and R. Winter, "Neural nets for adaptive filtering and adaptive pattern recognition," *in Computer*, 1988, vol. 21, no. 3, pp. 25-39.

[13] Iebeling Kaastra, Milton Boyd "Designing a neural network for forecasting financial and economic time series", *Neurocomputing*, 1996, Vol. 10, no. 3, pp. 215-236.

[14] Georg Dorffner, "Neural Network for Time Series Processing", *Neural Network World*, 1996, Vol. 6, pp. 447-468.

[15] Abbasi, R., M. Esmaeilpour, "Selecting Statistical Characteristics of Brain Signals to Detect Epileptic Seizures using Discrete Wavelet Transform and Perceptron Neural Network", *International Journal of Interactive Multimedia and Artificial Intelligence*, 2017, Vol. 4, no. 5, pp. 33-38.

Stephen Mack

Stephen Mack earned his BS in Electrical Engineering from The College of New Jersey and his MS in Electrical Engineering from the University of Virginia. His research interests include control theory, RF/communications, space systems, and machine learning.