1-1-2005

Experimental implementation of an artificial neural network-based active vibration control

Yong Xia
Ryerson University

Follow this and additional works at: <http://digitalcommons.ryerson.ca/dissertations>

Part of the [Mechanical Engineering Commons](#)

Recommended Citation

Xia, Yong, "Experimental implementation of an artificial neural network-based active vibration control" (2005). *Theses and dissertations*. Paper 403.

EXPERIMENTAL IMPLEMENTATION OF AN ARTIFICIAL NEURAL NETWORK-BASED ACTIVE VIBRATION CONTROL

by

Yong Xia

B.E. - Instrumentation

Tianjin University, China, 1992

A thesis

presented to Ryerson University

in partial fulfillment of the
requirements for the degree of
Master of Applied Science
in the Program of
Mechanical Engineering

Toronto, Ontario, Canada, 2005

© Yong Xia 2005

PROPERTY OF
RYERSON UNIVERSITY LIBRARY

UMI Number: EC53776

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform EC53776
Copyright 2009 by ProQuest LLC
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institution or individuals for the purpose of scholarly research.

Signature: _____

Date: 28/8/2005

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature: _____

Date: 28/8/2005

ABSTRACT

EXPERIMENTAL IMPLEMENTATION OF AN ARTIFICIAL NEURAL NETWORK-BASED ACTIVE VIBRATION CONTROL

Yong Xia

Master of Applied Science, 2005

Mechanical Engineering, Ryerson University

Vibration control strategies strive to reduce the effect of harmful vibrations on machinery and people. In general, these strategies are classified as passive or active. While passive vibration control techniques are generally less complex, there is a limit to their effectiveness. Active vibration control strategies, on the other hand, require more complex algorithms but can be very effective. In this current work, a novel active vibration control experimental system, including the hardware setup and software development environment, has been successfully implemented. A static artificial neural network-based active vibration control system has been designed and tested based on the experimental system. The artificial neural network is trained to model the plant using a backpropagation algorithm. After training, the network model is used as part of a feedforward controller. The efficiency of this controller is shown through experimental tests.

ACKNOWLEDGEMENTS

The author would first and foremost like to thank his thesis supervisor, Dr. Ahmad Ghasempoor, for his guidance and support in this research.

The author would also like to extend thanks to Mr. Roy Churaman, Mr. Devin Ostrom and Mr. Joseph Amankrah for their help with preparing some experimental hardware.

Last but not least, the author would like to thank the Department of Mechanical and Industrial Engineering and the School of Graduate Studies at Ryerson University for their support in terms of department facilities, research stipend and scholarship.

TABLE OF CONTENTS

AUTHOR'S DECLARATION.....	iii
ABSTRACT.....	v
ACKNOWLEDGEMENTS.....	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	xi
LIST OF FIGURES	xii
LIST OF APPENDICES.....	xiv
NOMENCLATURE	xv
CHAPTER 1 INTRODUCTION.....	1
1.1 Vibration Control.....	1
1.1.1 Passive Methods.....	1
1.1.2 Active Methods.....	2
1.1.3 Hybrid Methods.....	3
1.2 Objective.....	4
CHAPTER 2 LITERATURE REVIEW.....	5
2.1 Active Vibration Control.....	5
2.2 Control Strategies.....	7
2.2.1 Feedforward Control.....	8
2.2.2 Feedback Control.....	9
2.2.3 Feedforward vs. Feedback.....	9
2.3 Control System Design.....	10

2.3.1	Linear Filter Architecture/Algorithm Combinations.....	10
2.3.1.1	Adaptive Filters.....	10
2.3.1.1.1	FIR Filters.....	10
2.3.1.1.2	IIR Filters.....	11
2.3.1.1.3	FIR Filters vs. IIR Filters.....	12
2.3.1.2	Adaptation Algorithms.....	12
2.3.1.2.1	FIR Filters / LMS Algorithm Combination.....	13
2.3.1.2.2	FIR Filters / Filtered-x LMS Algorithm Combination.....	14
2.3.2	Nonlinear Architecture /Algorithm Combinations.....	16
2.4	Artificial Neural Network.....	17
2.4.1	Multilayer Feedforward Neural Network.....	18
2.4.2	Backpropagation Algorithms.....	20
2.4.3	Learning Methods.....	21
2.4.4	Training Styles.....	22
2.4.5	Normalization.....	22
2.4.6	Improving Generalization.....	22
2.4.7	Dynamic Neural Network.....	24
2.5	ANN Applications in AVC.....	24
2.5.1	Batch-Training Dynamic ANN-Based AVC.....	25
2.5.2	On-Line-Training Dynamic ANN-Based AVC.....	28
2.6	Real-Time Concepts for Dynamic ANN-Based AVC.....	32
2.6.1	Real-Time Performance.....	33
2.6.2	Real-Time Operating Systems.....	33

2.6.3 Real-Time Control.....	33
2.6.4 Real-Time Signal Processing.....	33
2.7 Challenges in Current AVC Applications.....	34
CHAPTER 3 METHODOLOGY.....	35
3.1 ANN-Based AVC Strategy.....	35
3.2 Simulated Annealing.....	39
3.3 Resilient Propagation.....	41
3.4 ANN Model Design.....	42
CHAPTER 4 EXPERIMENTAL SETUP.....	44
4.1 Hardware Setup.....	44
4.2 Software Environment.....	47
4.3 Signal Processing.....	49
4.3.1 Sampling Time.....	49
4.3.2 Window Functions.....	49
4.3.3 Sampling Rate.....	50
4.3.4 Magnitude Scales in FFT.....	50
4.4 Bending Vibration Analysis of the Beam.....	51
4.5 Natural Frequencies of the Overall System.....	55
4.6 Introducing Nonlinearity to the System.....	57
CHAPTER 5 EXPERIMENTAL IMPLEMENTATION.....	59
5.1 AVC System Design.....	59
5.2 ANN Training Data.....	60
5.3 Normalization.....	65

5.4	ANN Architecture.....	65
5.5	Training Algorithms.....	66
5.6	Results and Discussion.....	67
5.7	Modified AVC System.....	70
5.8	Results and Discussion of the Modified AVC System.....	74
CHAPTER 6 CONCLUSIONS AND FUTURE WORK.....		81
6.1	Contribution.....	81
6.2	Conclusions.....	81
6.3	Future Work.....	81
APPENDICES.....		83
A.	DAQ Board Specifications.....	84
B.	Proximity Sensor Specifications.....	86
REFERENCES.....		87

LIST OF TABLES

Table 5-1: Experimental design example for collecting 50-set data.....	63-64
--	-------

LIST OF FIGURES

Figure 2-1: Active control of sound waves in a duct.....	6
Figure 2-2: Principle of feedforward control.....	8
Figure 2-3: Principle of feedback control.....	9
Figure 2-4: Transversal FIR Filter structure.....	11
Figure 2-5: A generic IIR digital filter.....	12
Figure 2-6: Adaptive filter system with a controller based on the LMS.....	13
Figure 2-7: Adaptive filter system with a controller based on the filtered-x LMS algorithm.....	15
Figure 2-8: An example of the structure of a multilayer feedforward neural network.....	18
Figure 2-9: Early stopping method.....	23
Figure 2-10: Neural network identifier and Neural Network Controller in [24].....	26
Figure 2-11: Training of Neural Network Controller (NNC) in [24].....	26
Figure 2-12: Inverse system identification in [23].....	27
Figure 2-13: Direct inverse control of a plant in [23].....	28
Figure 2-14: Commonly used feedforward active control arrangement.....	29
Figure 2-15: Two multilayer feedforward neural networks used in [5].....	30
Figure 3-1: Feedforward AVC strategy.....	36
Figure 3-2: Training of the ANN model.....	37
Figure 3-3: Static ANN-based AVC methodology.....	38

Figure 3-4: Simulated Annealing minimization algorithm.....	40
Figure 4-1: Hardware setup for implementing AVC design.....	45
Figure 4-2: Schematic diagram of experimental setup.....	45
Figure 4-3: Photographs of the experimental hardware setup	46-47
Figure 4-4: An example of user interfaces used in the current work.....	48
Figure 4-5: FFT graphs of a same signal with different sampling time.....	49
Figure 4-6: FFT graphs of a same signal on a linear scale and a dB scale.....	51
Figure 4-7: A cantilever beam.....	52
Figure 4-8: The first three mode shapes of the cantilever beam.....	53-54
Figure 4-9: Measuring the system natural frequencies via experiments.....	55-56
Figure 4-10: Two ways to introduce nonlinearity.....	57
Figure 4-11: Spectrum comparison between driving the control shaker soft and hard.....	58
Figure 5-1: The proposed AVC system.....	60
Figure 5-2: General model of a process or system.....	61
Figure 5-3: ANN architecture of the proposed AVC system.....	66
Figure 5-4: Performance of the proposed AVC system.....	68-69
Figure 5-5: The modified AVC system.....	72
Figure 5-6: ANN architecture of the modified AVC system.....	73
Figure 5-7: Performance of the modified AVC system.....	75-76
Figure 5-8: Performance examples of the modified AVC system.....	77-80

LIST OF APPENDICES

Appendix A: DAQ Board Specifications 84-85

Appendix B: Proximity Sensor Specifications..... 86

NOMENCLATURE

x	input signal
y	output signal
e	error signal
d	disturbance signal
p	primary disturbance
u	control disturbance
w	weight
b	bias
v	activation potential
t	target output
μ	adaptation parameter/step-size/convergence factor
Z^{-1}	a unit sample of delay
E	error criterion/total error/cost function
α	learning rate parameter or step size in ANN
$\lambda_{\max}$	maximum curvature, or the maximum eigenvalue of Hessian matrix
c_i^*	coefficients of the estimated FIR filter model of the forward path
ζ	damping ratio
ω	natural frequency radians/sec
f	natural frequency in Hz
x_{scaled}	normalized value
$x_{\min}$	minimum value

$x_{\max}$	maximum value
V1	signal amplitude of the primary shaker
V2	signal amplitude of the control shaker
F1	signal frequency of the primary shaker
F2	signal amplitude of the control shaker
PD	phase difference between the signals of the two shakers
dB	decibel
AVC	active vibration control
DSP	digital signal processing
FIR	finite impulse response
IIR	infinite impulse response
LMS	least mean square
MSE	mean square error
MS	mega-sample
RMS	root mean square
ANN	artificial neural network
TDL	tapped delay line
NNI	neural network identifier
NNC	neural network controller
A/D	analog-to-digital
D/A	digital-to-analog
ADC	analog-to-digital converter
DAC	digital-to-analog converter

FFT	fast Fourier transform
DAQ	data acquisition
NI	National Instruments Co.
VI	virtual instrument
FEM	finite element methods
RPROP	resilient propagation
PID	proportional-integral-derivative

CHAPTER 1

INTRODUCTION

1.1 Vibration Control

Vibration is ever-present. Vibration control is the effort to reduce the negative effects of vibration effectively.

Two main groups of vibration control methods are passive and active methods [1] [2]. Other vibration control methods, e.g. semi-active and other hybrid methods, can be considered as a combination of passive and active methods.

1.1.1 Passive Methods

Passive vibration control methods include elimination of additional energy sources, eliminating or decreasing input forces and isolation from external disturbances [1]. Examples include balancing, reduction of mass, substitution of rolling bearings by slide ones, etc. The parametric modifications lead mainly to changes of mass and stiffness elements. The structural modifications deal with introducing additional constraints to the system or tearing of existing ones (i.e., continuity interruption of vibrator structure via introducing of intermediate elements), or addition of vibration eliminators [1]. Damping is also an important parameter but usually takes secondary consideration. It refers to mechanical energy dissipation that is converted to heat [1]. Therefore, it may cause the decrease of general working efficiency in some areas, e.g., for machines. The ideal device should run with minimal damping values. In cases when undesirable vibrations cannot be eliminated via constructional or parameter changes, damping will be introduced. Additional damping is usually provided by materials with appropriate damping values, frictional joints, additional dampers, etc.

Unfortunately, the use of passive vibration control methods is restricted because of small effectiveness in the range of low frequencies, sensitivity-dependent on application conditions, lack of robustness, increased size and weight, etc. On the other hand, these methods allow dissipating a great deal of vibration energy in the range of sufficiently high frequencies (energy dissipation increases with decrease of passive elements stiffness) [1] [2].

Better vibration control results may be achieved by using active methods.

1.1.2 Active Methods

Active vibration control (AVC) methods work by providing an additional energy supply to vibration systems. These methods alleviate the problems of contradictory requirements imposed on passive vibration control techniques, such as efficiency of device operation, low frequency vibration, dynamic stability, stiffness, etc. The additional energy supply can produce a force that compensates the forces that account for vibrations. To reduce the vibration, the additional energy should have appropriate power, frequency and phase, relative to existing vibration forces. AVC methods can change the system parameters in an active way. Some commonly utilized AVC methods include adaptive filter control, Proportional-Integral-Derivative (PID) control, fuzzy logic control and artificial neural network (ANN)-based control. As the vibration control method used in this research, the AVC method will be introduced in detail in chapter 2.

AVC is a highly multidisciplinary field with elements from structural dynamics, signal processing, materials science, mechanical vibration, actuator and sensor technology. The applications of AVC are also diverse. The following are some example areas for AVC application.

Chatter is a machining process instability resulting from self-excited vibration caused by the interaction of the chip removal process and the structure of the machine tool. Chatter occurs especially when machining flexible parts such as turbine blades, or when machining deep features using slender end mills. Chatter can limit metal removal

rates (productivity), cause poor surface finish and irritative noise, and accelerate tool wear. There has been a large research effort concerned with the suppression of chatter. Applying ANN-based AVC in machining chatter suppression is a future work of the research presented in this thesis.

Active noise control is an attractive means to achieve large amounts of noise reduction in a small package, particularly at low frequencies. Current applications include: control of aircraft interior noise by use of lightweight vibration sources on the fuselage and acoustic sources inside the fuselage; reduction of helicopter cabin noise by active vibration isolation of the rotor and gearbox from the cabin; attenuation of unavoidable noise in automobiles (electronic mufflers, engine mounts, and so on).

AVC can be used for suppressing the vibrations within a building generated by high winds or an earthquake, for reduction of sway in tall buildings and vibration of construction machinery, and to prevent large bridges from damage by flutter [1] [2].

AVC has been used in vibration-sensitive machines. Examples include surgical microscopes, electronic equipment, lasers, MRI units, scanning electron microscopes, and computer disk drives.

AVC can also be used in transportation equipment, life sciences, marine, semiconductor industry, and many other areas as well.

1.1.3 Hybrid Methods

There are other vibration control methods that can be considered as a combination of passive and active methods. One of these methods is called semi-active vibration control and includes passive elements where elastic and damping forces may be changed depending on the conditions. Such methods require an external energy source with much lower power requirements.

Another hybrid method uses both active and passive elements. The active part operates in the low frequency range and the passive one in the higher frequency range.

Considering the superior capabilities of active methods over passive methods, in

this research, AVC will be utilized. However, there are some technical challenges associated with the implementation of AVC. For example, adaptive filter control and PID control can only work effectively for linear vibration problems; most implemented artificial neural network (ANN)-based control methods rely on the plant output as the feedback signal and therefore have to solve the time-delay problem, which leads to the complexity of the control systems (see chapter 2 for details). This AVC research is working on dealing with these challenges.

1.2 Objective

The objective of the current work is to develop an effective adaptable AVC system. The system should be simple to implement, noise tolerant and capable of dealing with nonlinearities.

CHAPTER 2

LITERATURE REVIEW

2.1 Active Vibration Control

The research in active vibration control (AVC) has been expanding since Lueg's work in 1930s [6], and especially rapidly in the past three decades.

AVC is achieved by using a control source to introduce a secondary (control) disturbance into a system to cancel the existing (primary) disturbance, thus resulting in an attenuation of the original vibration [3]. These secondary sources are interconnected through an electronic system using a specific signal-processing algorithm for a particular cancellation scheme. To explain the concept of AVC, a classical application of active noise control is always used as an example [4]. The principle of destructive interference used is not only limited to the control of acoustic waves, but also successfully applied to the control of other vibration.

The classical application is active control of sound waves in a small duct, which is shown in Figure 2-1. An actuator (a loudspeaker) and two sensors (two microphones in this case) are used. Sensor A, which is called a reference sensor, is used to measure the advanced information on the disturbance sound wave that propagates in the duct, and sensor B, which is called an error sensor, is used to monitor the performance of the active sound control system, thus providing feedback to a control algorithm. To keep this example simple, sensor A is assumed to be not coupled with the actuator, so sensor A only measures the disturbance sound field. The control structure of Figure 2-1 is called "feedforward", because the controller feeds the actuator with a signal based on the advanced information obtained from sensor A. If the controller works properly, the signal sent to the actuator will generate a sound wave, which will cancel the disturbance

sound wave at the location of sensor B. Because only plane wave propagation is considered in such a small duct commonly, the sound field will be uniform in any section of the duct, and the sound will be reduced from sensor B to the end of the duct. Details of active sound control theory and applications can be found in [4].

Comparing with AVC, passive vibration control methods suppress vibration by using energy absorbing dampers to consume energy input or by changing the system structures or conditions to reduce the energy inputs or generated energy.

Figure 2-1: Active control of sound waves in a duct [4]

The concept of AVC has been known for more than 60 years. The basic ideas of AVC were proposed in 1936, when Paul Lueg first described the design of active noise control in a patent published in the United States. Even though the concept is simple, it is only with the development of low-cost fast digital signal processing (DSP) systems during the last 20 years that the implementation of practical active sound and vibration control systems has become feasible. It is desirable for the vibration canceller to be digital, where signals from electroacoustic or electromechanical transducers are sampled and processed in real time using DSP systems. Digital technology is well suited for adaptive control systems or control configurations where a lot of precision is required in order to achieve good performance.

The continuous progress of AVC involves the development of improved adaptive signal processing algorithms, transducers, and DSP hardware. Since the characteristics

of the vibration source and the environment are time varying, the frequency content, amplitude, phase, and vibration velocity of the undesired vibration are not stationary. An AVC system must therefore be adaptive in order to cope with these variations. AVC systems can be used in linear or nonlinear control problems.

For linear control problems, adaptive filters combined with some algorithms are always used. Adaptive filters adjust their coefficients to minimize an error signal and can be realized as finite impulse response (FIR), infinite impulse response (IIR), lattice, and transform-domain filters [6]. The FIR filter is also called a transversal filter and the IIR filter is also called a recursive filter. The algorithms used for adaptive filters are generally based on gradient descent algorithms. Examples are least-mean-square (LMS) and filtered-X LMS algorithms. For nonlinear control problems, many researchers have focused on neural networks combined with some specific algorithms, such as the standard gradient descent backpropagation algorithm. Moreover, neural networks proved experimentally to be robust for not only nonlinear control but also linear control [7]. Some commonly used architectures and algorithms in AVC will be introduced in a latter section.

More sophisticated algorithms allow faster convergence and greater vibration suppression and are more robust with respect to interference. A good review of the different control techniques that have been used for the active control of sound and vibration can be found in [6] [8] [11] [13] and [15]. The development of improved DSP hardware has allowed these more sophisticated algorithms to be implemented in real-time to improve system performance.

2.2 Control Strategies

Basically, in the area of AVC, there are two kinds of control strategies:

feedforward and feedback [3] [4] [6].

2.2.1 Feedforward Control

The noise suppression example in the preceding section is a feedforward control strategy application. The principle of feedforward control is presented in Figure 2-2. Feedforward controllers rely on the availability of a reference signal correlated to the primary disturbance. This signal is passed through an adaptive controller. The output of the controller is applied to the system by secondary sources. The filter coefficients are adapted in such a way that the error signal at one or several critical points is minimized. The idea is to produce a secondary disturbance such that it cancels the effect of the primary disturbance at the location of the error sensor.

Figure 2-2: Principle of feedforward control [30]

One important point to note is that the signal must be received by the controller in sufficient time for the required control signal to be generated and output to the control/secondary source when the disturbance (from which the reference signal was generated) arrives. Systems, for which the active control system produces the control signal at the downstream location at the same time that the primary signal arrives, are referred to as “causal”. Causality is a condition that all feedforward designs must satisfy if the vibration to be controlled is not periodic [6]. As will be discussed in section 2.3, the reason why most transfer function models utilize tapped delay lines is to enable the modeling of the explicit system time delays to maintain causality within the control

schemes. If the vibration to be controlled is periodic, it is possible to get a similar outcome without satisfying causality if the control signal output and the arriving disturbance have a phase difference equal to an integer number of periods.

Another point to note is that there is no guarantee that the global response is also reduced at other locations and, unless the response is dominated by a single mode, there are places where the response can be amplified; the method can therefore be considered as a local one, in contrast to feedback which is global (as will be discussed in the following section).

2.2.2 Feedback Control

The principle of feedback control is presented in Figure 2-3. The output y of the system is compared to the reference input x and the error signal, $e = x - y$, is passed into a compensator $H(s)$ and applied to the system $G(s)$. The design problem consists of finding the appropriate compensator $H(s)$ such that the closed-loop system is stable.

Figure 2-3: Principle of feedback control [30]

2.2.3 Feedforward vs. Feedback

Generally, a feedforward control system should be implemented whenever it is possible to obtain a suitable reference signal because of its inherent stability characteristics and usually superior performance to a feedback control system [3] [6]. The use of feedback control has been popular in the area of structural vibration because of its ability to damp structural vibrations without the need to be able to measure a reference signal in advance [3] [4] [6] [8].

2.3 Control System Design

Many of the electronic systems utilized in control schemes derive control inputs for secondary sources (as in Figure 2-2) via modified adaptive signal processing architecture/algorithm combinations. Since the characteristics of the input signal and system response are unknown, or may be slowly changing with time, the controller must “learn”, and be able to “relearn” to cope with changes in the signal structure. Basically, there are two kinds of such architecture/algorithm combinations: linear filter combinations and nonlinear combinations.

2.3.1 Linear Filter Architecture/Algorithm Combinations

Many of today’s implementations of active control use adaptive linear filtering techniques. With the advances in digital technology over the past several decades, adaptive DSP has become a firmly established field, encompassing a wide range of applications. One of the most common forms of adaptive architecture/algorithm linear filter combinations is the filter-based controller adapted by using Least Mean Square (LMS)-based algorithms (discussed in section 2.3.1.2) [8]. The aim of the adaptation algorithm is to adapt the filter coefficients such that the error signal is minimized.

2.3.1.1 Adaptive Filters

Adaptive filters can be realized as (transversal) finite impulse response (FIR), (recursive) infinite impulse response (IIR), lattice, and transform-domain filters [8]-[13] [15]. Two primary types of digital filters used in (DSP) applications are FIR filters and IIR filters [8].

2.3.1.1.1 FIR Filters

Figure 2-4 shows the structure of a transversal FIR filter with N taps adjustable weights. The Z^{-1} block represents a unit sample of delay. The impulse response is “finite” because there is no feedback in the filter. The FIR filter is obtained by

combining a tapped delay line with a linear network [8] [13].

Figure 2-4: Transversal FIR Filter structure [8]

The tap-weight vector, $w(n)$, is represented as $w(n) = [w_0(n), w_1(n), \dots, w_{N-1}(n)]^T$; the tap-input vector, $x(n)$, is represented as $x(n) = [x(n), x(n-1), \dots, x(n-(N-1))]^T$; the FIR filter output, $y(n)$, can then be expressed as

$$y(n) = \sum_{i=0}^{N-1} w_i(n)x(n-i) \quad (2-1)$$

where n is the time index and N is the order of the filter.

2.3.1.1.2 IIR Filters

Another primary digital filter used in DSP application is IIR. The impulse response is "infinite" because there is feedback in the filter. An IIR filter produces an output, $y(n)$, which is the weighted sum of the current and past inputs, $x(n)$, and past outputs. Figure 2-5 is a generic IIR digital filter and equation (2-2) shows how to calculate the output of the filter [31].

$$y(n) = g[x(n) + a_1 x(n-1) + \dots + a_N x(n-N) + b_1 y(n-1) + \dots + b_M y(n-M)] \quad (2-2)$$

If the generic IIR filter in Figure 2-5 does not operate on the past values of the output, i.e., it would only have non-zero a_i coefficients in the above equation, but all b_j coefficients would be zero, then it changes to a FIR filter as shown in Figure 2-4.

Figure 2-5: A generic IIR digital filter [31]

2.3.1.1.3 FIR Filters vs. IIR Filters

The FIR filter and The IIR filter have their separate advantages and disadvantages. Overall, though, the advantages of the FIR filter outweigh its disadvantages, so it is used much more than the IIR filter [8]. Compared to the FIR filter, the IIR filter can achieve a given filtering characteristic using less memory and calculation than a similar FIR filter. However, compared to IIR filters, FIR filters have more advantages as listed below [32]:

- They can easily be designed to be "linear phase";
- They are simple to implement;
- They are suited to multi-rate applications;
- They have desirable numeric properties; and
- They can be implemented using fractional arithmetic.

2.3.1.2 Adaptation Algorithms

There are many possible adaptation algorithms, and one of the most useful algorithms is the LMS algorithm [8] [10] [11]. Many LMS-based algorithms and their variants exist. This section focuses on the LMS algorithm and an LMS-based algorithm, i.e., the filtered-x LMS algorithm for FIR filters.

2.3.1.2.1 FIR Filters / LMS Algorithm Combination

Figure 2-6 shows a block diagram of an adaptive filter system identification model. The unknown system is modeled by an FIR filter with adjustable coefficients. Both the unknown system and the FIR filter model are excited by an input sequence $x(n)$. The adaptive FIR filter output $y(n)$ is compared with the unknown system output $d(n)$ to produce the error signal $e(n)$. The error signal represents the difference between the unknown system output and the model output. The error $e(n)$ is then used as the input to an adaptive control algorithm, which corrects the individual tap weights of the filter. This process is repeated through some iterations until the error signal $e(n)$ becomes sufficiently small. The resultant FIR filter response now represents that of the previously unknown system.

Figure 2-6: Adaptive filter system with a controller based on the LMS [11]

The LMS algorithm adjusts the weights and biases of the FIR filter so as to minimize the mean square error. LMS algorithm is an example of supervised training, in which the learning rule is provided with a set of examples of desired network behavior: $\{x_1, t_1\}, \{x_2, t_2\}, \dots, \{x_q, t_q\}$. Here x_q is an input to the network, and t_q is the corresponding target output. As each input is applied to the network, the network output is compared to the target. The error is calculated as the difference between the target output and the network output. The Mean Square Error (MSE) to be minimized is

$$MSE = \frac{1}{Q} \sum_{k=1}^Q e(n)^2 = \frac{1}{Q} \sum_{k=1}^Q (t(n) - y(n))^2 \quad (2-3)$$

The LMS algorithm is initialized by setting all the weights to zero at time $k=0$. Tap weights and bias are updated using the relationship [8] [13]

$$w(k+1) = w(k) + 2 \mu e(k)x(k) \quad (2-4)$$

$$b(k+1) = b(k) + 2 \mu e(k) \quad (2-5)$$

where $w(k)$ represents the tap weights of the transversal filter, $e(k)$ is the error signal, $x(k)$ represents the tap inputs, $b(k)$ is the bias, and the factor μ is a convergence factor (also called the learning rate or the step length) whose value influences the amount by which the weight vector is altered at each iteration. To ensure convergence, μ should satisfy the condition [13]:

$$0 < \mu < (1 / \lambda_{\max}) \quad (2-6)$$

where $\lambda_{\max}$ is the maximum eigenvalue of the input correlation matrix.

2.3.1.2.2 FIR Filters / Filtered-x LMS Algorithm Combination

Another popular adaptive architecture/algorithm combination is the FIR filter/filtered-x LMS algorithm combination. The filtered-x LMS algorithm is also called the multi-error LMS algorithm or multi-channel LMS algorithm, which is an extension of the LMS algorithm.

In Figure 2-6, the output of the FIR filter is an estimate of the unknown system. However, in real control applications, the adaptive filter works as a controller controlling dynamic systems, which contain actuators, amplifiers, etc. The estimate in this case can thus be seen as the output signal from a dynamic system, i.e., a forward path as in Figure 2-7. Since there is a dynamic system between the filter output and the estimate, the direct LMS algorithm is likely to be unstable in this application due to the phase shift (delay) introduced by the forward path [4]. In this case, a model of the forward path can be

introduced to filter the reference signal to the adaptive algorithm as in Figure 2-7. The compensated adaptive algorithm obtained is the filtered-x LMS algorithm [10].

Figure 2-7: Adaptive filter system with a controller based on the filtered-x LMS algorithm [10]

As shown in section 2.3.1.1.1, the FIR filter output, $y(n)$, can be expressed as equation (2-1). Equation (2-1) can also be written as

$$y(n) = w^T(n)x(n) \quad (2-7)$$

where

$$x(n) = [x(n), x(n-1), \dots, x(n-N+1)]^T \quad (2-8)$$

is the input signal vector to the adaptive filter and

$$w(n) = [w_0(n), w_1(n), \dots, w_{N-1}(n)]^T \quad (2-9)$$

is the adjustable filter coefficient vector. In control applications, the estimation error $e(n)$ is defined by the difference between the desired signal (desired response) $d(n)$ and the output signal from the forward path or plant under control $y_c(n)$:

$$e(n) = d(n) - y_c(n) \quad (2-10)$$

$$x_{c^*}(n) = \begin{bmatrix} \sum_{i=0}^{I-1} c_i^* x(n-i) \\ \sum_{i=0}^{I-1} c_i^* x(n-i-1) \\ \vdots \\ \sum_{i=0}^{I-1} c_i^* x(n-i-N+1) \end{bmatrix} \quad (2-11)$$

where n is the time index, i is the order and c_i^* is the coefficients of the estimated FIR filter model of the forward path [10].

$$h_{c^*}(n) = \begin{cases} c_n^* \rightarrow n \in \{0, \dots, I-1\} \\ 0 \rightarrow \text{else} \end{cases} \quad (2-12)$$

where $h_{c^*}(n)$ is the estimated FIR filter model of the forward path.

$$w(n+1) = w(n) + \mu \cdot x_{c^*}(n)e(n) \quad (2-13)$$

where μ is a convergence factor (also called learning rate or step length).

In order to ensure that the filtered-x LMS algorithm is stable, the maximum value for the convergence factor μ should be given approximately by [8]:

$$\mu < \frac{2}{NE[x^2(n)]} \quad (2-14)$$

where $E[x^2(n)]$ is the mean square value of $x(n)$.

2.3.2 Nonlinear Architecture /Algorithm Combinations

Even though the combination of a transversal filter using modified LMS algorithms has been widely demonstrated to be very useful, it has a potential limitation because it is designed for linear control problems. In other words, the control signal, as well as the associated measured error signal used in the adaptation process, must be linear functions of the reference signal used by the adaptive filter to derive the control signal [6]. The linear filter architectures may not perform well in cases where

nonlinearities are found in an active control system. One of the most common sources of nonlinearity in the field of active control of vibration is the actuator. An actuator has typically a nonlinear response when it operates with an input signal having an amplitude close to (or above) the nominal input signal value, or when it operates at a frequency outside of the normal frequency range of operation (or close to the limits) [20]; a control actuator also may have some nonlinear performance characteristic, such as where it generates some harmonics and introduces the harmonics into the system, which must be compensated for; and where a power based (intensity) error signal is used, which will be twice the frequency of the reference signal [6], for example, when some band-pass filters are used for the error signal. Another example of nonlinearity in active vibration control is when a sinusoidal reference signal is used to derive a signal to control a disturbance containing both the reference tone and several harmonics. Nonlinear behaviors can also occur when the dynamics of the system to be controlled are nonlinear.

Therefore, what is desired in these situations is a nonlinear controller, which can improve the control performance of a system associated with some form of nonlinearity. One such controller arrangement, which has received increased attention in recent years, is the artificial neural network (ANN). ANN-based AVC is the focus of this research, so ANN architectures and algorithms, and ANN application in AVC will be introduced in detail.

2.4 Artificial Neural Network

An ANN is a system whose architecture is inspired by the arrangement of nerves in biological systems and by their operation. An ANN is an extensively parallel interconnection of simple neurons that has the ability to learn from its environment and store the acquired knowledge for future use [13]. ANNs are used for pattern recognition

or function approximation. In AVC, ANNs are mostly used for their function approximation capability. Properly designed and trained neural networks are capable of approximating any linear or nonlinear function to the desired degree of accuracy [13]. A strong case can be made that neural network implementation is simply a form of multivariate statistical analysis [6].

There are a variety of design and learning techniques to choose from when using an ANN. The pattern of connections between the neurons, called network architecture, and the method in which the weights of the connections are determined, known as learning algorithm, are the elements that characterize an ANN.

2.4.1 Multilayer Feedforward Neural Network

The most common ANN architecture used in AVC is the multilayer feedforward neural network. Figure 2-8 illustrates an example of the structure of a multilayer feedforward neural network, which comprises a layer of input signals, 2 hidden layers (layer 1 and layer 2) of neurons, and an output layer (layer 3) of neurons.

Figure 2-8: An example of the structure of a multilayer feedforward neural network [36]

A multilayer feedforward neural network (see Figure 2-8) comprises a layer of input signals, one or more hidden layers of neurons, and an output layer of neurons. A layer consists of a single or multiple neurons. Neurons in each layer are connected to all

neurons in adjacent layers. The network in Figure 2-8 is of the feedforward type, wherein the effects of the input signals are propagated through the networks layer by layer. Differences between the desired outputs (targets) and the network outputs are the “errors”. The connection strengths (‘weights’) and ‘biases’ are updated during training (learning) such that the network produces the desired output for the given input. The multilayer feedforward neural networks trained with a back-propagation algorithm are compact and provide excellent generalization (i.e., accurate outputs for inputs not encountered during training) [13]. Mathematically, neuron j having m inputs is described as follows [5] [13]:

$$y_j = f_j(v_j)$$

$$v_j = \sum_{i=1}^m w_{ji} x_i + b_j \quad (2-15)$$

where x_i are input signals, w_{ji} are weights from neuron i to neuron j , b_j is bias, v_j is the activation potential, f_j is the activation/transfer function and y_j is the output signal of the neuron. The bias applies an affine transformation to the linear combination of inputs and weights. The type of activation function used determines the neuron output for the given weights, inputs and bias. For example, the following activation functions are always used [13]:

$$f_j(v_j) = v_j, \quad \text{linear}$$

$$f_j(v_j) = \frac{1}{1 + e^{-v_j}}, \quad \text{log-sigmoid}$$

$$f_j(v_j) = \tanh(v_j) = \frac{e^{v_j} - e^{-v_j}}{e^{v_j} + e^{-v_j}}, \quad \text{tan-sigmoid} \quad (2-16)$$

Feedforward networks often have one or more hidden layers of sigmoid neurons followed by an output layer of linear neurons. Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear and linear relationships between input and output vectors. The linear output layer lets the network produce values outside

the range -1 to +1. However, if the outputs of a network need to be constrained (such as between 0 and 1), the output layer should use a sigmoid transfer function (such as log-sigmoid).

Theoretically, a two-hidden-layer network having a sigmoid first hidden layer and a linear second layer can be trained to approximate most functions (linear or nonlinear) arbitrarily well [13].

2.4.2 Backpropagation Algorithms

There are many variations of the backpropagation algorithm. The simplest one is the gradient descent algorithm, which updates the network weights and biases in the direction in which the performance function decreases most rapidly -- the negative of the gradient.

For a given set of inputs to the network, outputs are computed for each neuron in the first layer and forwarded to the next layer. The signals propagate on a layer-by-layer basis until the output layer is reached. The weights and biases remain unchanged during the “forward pass”. The output of the network is compared with the desired value (t_j), and the difference gives the error:

$$e_j = t_j - y_j \quad (2-17)$$

Applying the delta rule [13], which adjusts the weights so as to minimize the mean square error, the total error (or “error criterion” in some papers) is defined as:

$$E = \frac{1}{2} \sum_{j=1}^c e_j^2 \quad (2-18)$$

where c is the number of neurons in the output layer.

The error E represents the cost function, and the weights and biases are updated to minimize it. The backpropagation algorithm minimizes the cost function in a manner similar to the steepest descent method [13]. The computed partial derivatives (sensitivity) $\partial E / \partial w_{ji}$ determine the search direction for updating the weights w_{ji} as

$$w_{ji}(k+1) = w_{ji}(k) - \alpha \frac{\partial E(k)}{\partial w_{ji}(k)} \quad (2-19)$$

where α is the learning rate parameter (step size), k is the current time and $(k+1)$ is the next time step. For stable learning,

$$\alpha < \frac{2}{\lambda_{\max}} \quad (2-20)$$

where $\lambda_{\max}$ is the maximum curvature, or the maximum eigenvalue of the Hessian matrix [13]. The weights and biases are updated during the “backward pass” starting from the output layer, and recursively computing the local gradient for each neuron.

The gradient descent algorithm can be refined using a ‘momentum term’ that has a stabilizing effect on the backpropagation algorithm [13]. By the use of momentum, a larger α can be used, while maintaining the stability of the algorithm. Momentum allows a network to respond not only to the local gradient, but also to recent trends in the error surface. It also allows the network to ignore small features in the error surface. Without momentum, a network may get stuck in a shallow local minimum. With momentum, a network can slide through such a minimum [13].

In practical application, the two backpropagation algorithms (gradient descent, and gradient descent with momentum) are often too slow. Modifications fall into two main categories [36]. The first category uses heuristic techniques, which were developed from an analysis of the performance of the standard steepest descent algorithm. One heuristic modification is the momentum technique, such as variable learning rate backpropagation, and resilient backpropagation [36]. The second category of fast algorithms uses standard numerical optimization techniques, such as conjugate gradient, quasi-Newton, and Levenberg-Marquardt [13].

2.4.3 Learning Methods

The learning methods include supervised learning and unsupervised learning. In supervised learning neural networks are adjusted, or trained, so that a particular input

leads to a specific target output. In unsupervised learning, there is not a specific target output. Unsupervised networks can be used, for instance, to identify groups of data. However, in AVC, supervised training methods are commonly used.

2.4.4 Training Styles

The training of a neural network is complete when the error (or change in the error) reduces to a predetermined small value. ANN training methods comprise batch training and incremental training. Batch training of a network proceeds by making weight and bias changes based on an entire set (batch) of input vectors. Incremental training changes the weights and biases of a network as needed after presentation of each individual input vector. Incremental training is sometimes referred to as “on-line” or “adaptive” training.

2.4.5 Normalization

Before training, it is often useful to scale the inputs and targets so that they always fall within a specified range [36]. Neural networks are very sensitive to absolute magnitudes. For this reason, the inputs and targets are usually scaled to give each of them equal importance and to prevent premature saturation of activation functions. All data to an ANN are normalized so that they correspond to roughly the same range of values. Normalization has the advantage of mapping the desired range of variable to a full working range.

2.4.6 Improving Generalization

One problem that occurs during neural network training is called overfitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations [36].

One method for improving network generalization is to use a network that is just large enough to provide an adequate fit [13]. Unfortunately, it is difficult to know

beforehand how large a network should be for a specific application. The standard ways to limit the capacity of an ANN including limiting the number of hidden units, limiting the size of weights and stopping the learning before it has time to overfit (early stopping).

In the early stopping method, the available data are divided into three subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second subset is the validation set. The error on the validation set is monitored during the training process. The validation error will normally decrease during the initial phase of training, as does the training set error. However, when the network begins to overfit the data, the error on the validation set will typically begin to rise (see Figure 2-9, which is an example drawn by using MATLAB). When the validation error increases for a specified number of iterations, the training is stopped, and the weights and biases at the minimum of the validation error are returned. The test set error is not used during the training, but it is used to compare different models. It is also useful to plot the test set error during the training process. If the error in the test set reaches a minimum at a significantly different iteration number than the validation set error, this may indicate a poor division of the data set [36].

Figure 2-9: Early stopping method

Another way recommended in [36] is Bayesian regularization. Bayesian regularization generally provides better generalization performance than early stopping, when training function approximation networks [36]. This is because Bayesian regularization does not require that a validation data set be separated out of the training data set. It uses all of the data. This advantage is especially noticeable when the size of the data set is small.

2.4.7 Dynamic Neural Network

Dynamic neural networks are always used to model dynamic systems [13]. A dynamic ANN has some time-delayed values of the outputs as its inputs by using tapped delay lines (TDL). A TDL is a delay line with at least one "tap". A delay-line tap extracts a signal output from somewhere within the delay line, optionally scales it, and usually sums with other taps from the signal output. A tap may be interpolating or non-interpolating. A non-interpolating tap extracts the signal at some fixed integer delay relative to the input. Tapped delay lines efficiently simulate multiple echoes from the same source signal.

One example of using TDL in a neural network model of a system is shown in Figure 2-10 (a). In this example each "Time Delay" operator yields a one-time-step delayed version of the input signal, and thereby builds a short-term memory into the system. This feature transforms a static ANN to a dynamic ANN whose output is a function of time [11] [12].

2.5 ANN Applications in AVC

The idea of using ANNs for nonlinear feedforward control problems was first proposed by K. S. Narendra and K. Parthasarathy [16], and then, the investigations into it by some other people have been reported (see, for example, [7], [19], [21], [22], [23]

and [24]). Generally, multilayer feedforward neural networks adapted using some extensions of the standard gradient descent backpropagation algorithm has received the most attention from the control community as a potential nonlinear filtering tool [6].

In the AVC area, the linear function is generally used for neurons in the output layer, and nonlinear functions (such as the sigmoid function) are used for neurons in hidden layers. The linear activation function used for output neurons can provide the control signals with the capacity to vary over the positive/negative range required for control; the nonlinear activation functions used for hidden neurons can provide nonlinear control.

2.5.1 Batch-Training Dynamic ANN-Based AVC

One example of batch-training dynamic ANN-based AVC is discussed in [24]. A Neural Network Identifier (NNI) and a Neural Network Controller (NNC) are used in this example (Figure 2-10). The NNI is a model of the system, which is a dynamic neural network and used to simulate the response and design the controller. The inputs to the NNI include excitation signals, control signals and two delayed values of plant output. The activation function is purely linear. The design of the NNC is based on the inversion of the plant model (NNI). The NNC has five hidden neurons and a single output neuron, which produces the controller voltages. The inputs consist of excitation signals, control signals and time-delayed target values. The hidden layer uses the tangent sigmoid activation function (as in equation 2-16), which limits the output to ± 1 for large values of the activation potential. This has a stabilizing effect on the controller signals. For the output neuron, the activation function is purely linear, which provides the control signals with the capacity to vary over the positive/negative range required for control.

Figure 2-10: Neural network identifier and Neural Network Controller in [24]

The significant feature in this example is the way to train the NNC. The NNI is trained in a batch mode, and then is used to train the controller through off-line simulation. An adaptive scheme was used for the controller, as shown in Figure 2-11.

Figure 2-11: Training of Neural Network Controller (NNC) in [24]

The problem in training the NNC is that no target values (in terms of control signals, which are the output of the NNC) are known that would minimize the tip acceleration (which is the goal). The significant idea of [24] to solve this problem is by defining the target as the sum of control signal and tip acceleration (NNI output, which is desired to be zero for vibration suppression). The error is the difference between the target value and the NNC predicted value (control signal). The error (the tip acceleration

in this case) is backpropagated to train the NNC. The NNC weights and biases are adjusted after every ten samples of data, which resulted in updated control signals. Once the NNC training was complete, it was connected to the plant to obtain controlled responses.

In another example, the simplest concept called “Direct Inverse Control” is implemented in an ANN-based AVC [23]. The principle of “Direct Inverse Control” is that if a process can be described by a function dependent on the past states of the system that may be dependent on past inputs and outputs, a network is trained as the inverse of the process since that plant can be approximated by an inverse function. The inverse model is subsequently applied as the controller for the process by inserting the desired output. [23].

To design a plant inverse model, input/output data must be collected for the operating range and input conditions of the system. To construct a direct inverse neural network controller, an ANN to model the inverse of the plant needs to be trained (see Figure 2-12). The inputs to the ANN inverse model are the state: $x(k)$ and the change in state: dx . The output is the input control signal going to an actuator or shaker.

Figure 2-12: Inverse system identification in [23]

After the ANN is trained, it is put into the direct inverse control framework. The input to the inverse plant model controller is the current state and the desired state. The output of the controller is the input control signal going to an actuator or shaker (see Figure 2-13, where D means one time step delay here).

Figure 2-13: Direct inverse control of a plant in [23]

Both of the above two examples demonstrate the efficiency and robustness of batch-training dynamic ANN-based AVC mechanism experimentally.

2.5.2 On-Line-Training Dynamic ANN-Based AVC

There has been an intensive interest in on-line-training dynamic ANN-base AVC since the idea of using ANN for nonlinear feedforward control problems was first proposed by K. S. Narendra and K. Parthasarathy ([7], [16], [19] and [20]). The commonly used feedforward AVC arrangements are similar to the one shown in Figure 2-14. “ANN 1” is a neural network controller and “ANN 2” is a neural network model of the plant. Reference signals are sent to “ANN 1”. Tapped delay lines are always used as the input layer in feedforward neural networks. Tapped delay lines yield time-step delayed versions of the input signal, and thereby built short-term memories into systems. This feature transforms a static network into a dynamic network whose output is a function of time (as discussed in section 1.4.7). Transfer function models utilize tapped delay lines to enable the modeling of the explicit system time delays to maintain causality within the control scheme (as discussed in section 1.2.1).

The reference signals are used by “ANN 1” (neural network controller) to derive a set of control signals. Each control signal is modified by some system dependent cancellation path transfer function, i.e., “ANN 2” (plant model), before sent to the plant through actuators. Each error signal is then the sum of the primary and control components (superposition of the signals in the plant environment) and measured by a sensor. The error signals are then used by the neural networks for weight adjusting.

The training of “ANN 2” can be done with classical neural networks algorithms, including backpropagation algorithms (with or without momentum), nonlinear optimization algorithms (quasi-Newton algorithms, conjugate gradient algorithms) or nonlinear identification techniques (nonlinear extended Kalman filtering or recursive-least-squares algorithms). A review of these algorithms can be found in [20]. The training of “ANN 1” can not be done with those classical algorithms, because of the tapped delay lines between the two neural networks. For this kind of on-line-training dynamic ANN-based AVC, finding the effective algorithms to train “ANN 1” has been a focus of many researches. The combination of multilayer feedforward neural networks adapted using some extensions of the standard gradient descent algorithm, together with the developments of those modified algorithms, can be found in the literature.

Figure 2-14: Commonly used feedforward active control arrangement

One example of such kind of on-line-training dynamic ANN-based AVC is shown in [7], in which two multilayer feedforward neural networks, each having a tapped delay line, are used; a modified gradient descent backpropagation algorithm for adapting the feedforward neural network which has a tapped delay line is developed, and the performance characteristics are assessed experimentally.

Figure 2-15 represents the two multilayer feedforward neural networks used in [7]. Figure 2-15 (a) is an ANN feedforward controller, i.e. “ANN 1” in Figure 2-14; and Figure 2-15 (b) is an ANN model of the cancellation path transfer function, i.e. “ANN 2” in Figure 2-14. In [7], the control achieved by using a 6x6x1 “ANN 1” (6

inputs, six nonlinear hidden layer nodes, and one linear output node), together with a 6x1 “ANN 2” (six inputs and one linear output node).

(a) An ANN feedforward controller

(b) An ANN model of the cancellation path transfer function

Figure 2-15: Two multilayer feedforward neural networks used in [7]

As discussed before, in applications of on-line-training dynamic ANN-based AVC, in order to derive an algorithm which will facilitate stable adaptation of an ANN-based feedforward active control system, the plant model is used to incorporate the frequency response characteristics of the control actuator and error sensor, as well as the response characteristics of the structural system which separates them, including delays due to the finite distance between the source and the sensor. In [7], this plant model is

modeled as a second neural network, the input of which is the control signals, $x_0(k)$, and the output of which is the feedforward control signals, $s(k)$, measured at the output of the error sensor. A reference input sample at time k , $x_{in}(k)$, which is in some way related to (but not necessarily linearly correlated with) an impending primary disturbance, $p(k)$, is used to derive the set of control signals, $x_0(k)$. Each error signal is then the sum (superposition) of the primary and control components [7]:

$$e_j(k) = p_j(k) + s_j(k) \quad (2-21)$$

The error criterion that the controller is to minimize is the sum of the mean square value of error signals from $N(e)$ error sensors [7]:

$$ErrorCriterion = \sum_{n=0}^{N_e-1} \xi_n(k) = \sum_{n=0}^{N_e-1} E\{e_n^2(k)\} \approx \sum_{n=0}^{N_e-1} e_n^2(k) \quad (2-22)$$

The gradient descent algorithm acts to adjust the weights of the control system [7]:

$$w(k+1) = w(k) - \alpha \Delta w(k) \quad (2-23)$$

The gradient estimate used in the adaptive algorithm is [7]:

$$\Delta w \approx \sum_{n=0}^{N_e-1} \frac{\partial e_n^2(k)}{\partial w} \quad (2-24)$$

The impediment to utilizing the standard backpropagation algorithm for adapting the controller neural network is the inclusion of the tapped delay line input to the transfer function model. The standard backpropagation algorithm cannot be used directly in this arrangement since it must backpropagate through a tapped delay line. Therefore, the standard gradient descent backpropagation algorithm must be modified to enable adaptation of the neural controller for use in feedforward control systems. In [7], steepest descent algorithms based on the filtered-x approach were introduced for the training of the controller network. In order to derive a modified algorithm, the error signals can first be backpropagated from the transfer function model output to the tapped

delay line if the transfer function model input uses the standard algorithm [7]. The outcome shows that it is in fact past and present versions of the nodal outputs which are used in updating the controller network weights, and not past and present values of the error signals (see [7] for details of the algorithm development).

One point should be stressed here is that the neural network controller and the cancellation path transfer function neural network are inherently different in their function, so they must be adapted separately [7]. Once converged, the cancellation path transfer function neural network is then simply used as a tool to facilitate stable adaptation of the neural network controller and is not modified itself in this process. The neural network controller is a “phase inverse” model, whose error signal is defined as the sum of its output and the signal whose inverse signal is desired (the superposition of the control signal and the primary disturbance in the structural domain); while the cancellation path transfer function neural network is a model, whose error signal is based on the difference between its output and some desired signal (the system response measured at the error sensor output to the control signal input) [7].

For this kind of on-line-training dynamic ANN-based AVC, finding the effective algorithms to train the ANN controller has been a focus of many researches. For example, in [19], an adjoint approach is introduced; and in [20], a heuristic procedure is introduced for the development of recursive-least-squares algorithms based on the filtered-x and the adjoint gradient approaches, which leads to the development of new recursive-least-squares algorithms for the training of the ANN controller.

2.6 Real-Time Concepts for Dynamic ANN-Based AVC

The following real-time concepts are very important for dynamic ANN-based AVC experimental implementation.

2.6.1 Real-Time Performance

The most common misconception associated with real-time performance is that it increases the execution speed of a program. While this is true in some cases, it actually enhances the application by providing more precise and predictable timing characteristics. With these enhancements, the exact time when certain events will occur can be determined [35]. Real-time performance can be achieved through either hardware or software.

2.6.2 Real-Time Operating Systems

A real-time system is one in which the correctness of the computations not only depends upon the logical correctness of the computation, but also upon the time at which the result is produced. If the timing constraints of the system are not met, system failure is said to have occurred [37]. Microsoft Windows 2000 and XP are general-purpose operating systems, but they are not real-time operating systems [37]. One of the main differences between real-time operating systems and general-purpose operating systems is the ability to guarantee a worst-case latency. On a general-purpose operating system, an external interrupt could be put into a queue and then serviced later after the operating system has finished its current operation and any other interrupts in the queue. On the other hand, a real-time operating system can halt its current process to handle an interrupt immediately. In essence, the real-time operating system guarantees event response within a certain interval [37].

2.6.3 Real-Time Control

With real-time control, a physical system can be monitored and simulated continually. Real-time control applications repeatedly perform a user-defined task with a specified time interval separating them [35].

2.6.4 Real-Time Signal Processing

Real-time signal processing has many of the same characteristics as real-time

control. It requires deterministic time intervals between repetitive events [35]. But instead of calculating a response, it performs signal processing on the acquired data. In dynamic ANN-based AVC applications, point-by-point analysis routines provide much better performance. Instead of analyzing blocks of data, these routines maintain a memory of previous data and calculate a new output based on the history of the data and the current value. Hard real-time performance is necessary in these systems because missing input values or even acquiring these values after a small time delay destroys the integrity of the historical data for future calculations.

2.7 Challenges in Current AVC Applications

There are some technical challenges associated with the current AVC applications. As introduced in section 2.3, adaptive filter control can only work effectively for linear vibration problems. As introduced in section 2.5, most implemented ANN-based control methods rely on the plant output as the feedback signal and therefore have to solve the time-delay problem, which leads to the complexity of the control systems (see section 2.5 for details).

The current work will try to deal with these challenges. The proposed methodology is introduced in the following chapter.

CHAPTER 3

METHODOLOGY

As stated previously, the objective of this research is to develop an effective, adaptable active vibration control (AVC) system. The system should be capable of dealing with nonlinearities, simple to implement and noise tolerant. Because of their many advantages, artificial neural networks (ANNs) are used to fulfill these requirements. The Methodology to design and implement such an AVC system will be presented in this chapter. First, the AVC strategy used in the current work will be presented. Then simulated annealing and resilient propagation algorithms used in the AVC system will be introduced. Finally, the general methods to design the ANN model will be discussed.

3.1 ANN-Based AVC Strategy

A feedforward control strategy (as introduced in section 2.2.1) is utilized in the current work. In a feedforward AVC strategy (see Figure 3-1), a controller relies on the availability of a reference signal correlated to the primary disturbance, i.e., $x(p)$ in Figure 3-1. This signal is passed through the controller. And then, the output of the controller, i.e., $x(u)$, is applied to the plant by a secondary source, i.e., Actuator in Figure 3-1. The plant output, i.e., Y in Figure 3-1, is the vibration response of the plant measured at some point.

Figure 3-1: Feedforward AVC strategy

In this feedforward AVC strategy, the following relation exists:

$$Y = f(x(p), x(u), t) \quad (3-1)$$

The controller is used for minimizing Y . In order to minimize Y , the control signal to the actuator, i.e., $x(u)$, can be determined through an optimization method such as steepest decent, where

$$x(u)^{k+1} = x(u)^k - \alpha^k \frac{\partial Y^k}{\partial x(u)^k} \quad (3-2)$$

where $\frac{\partial Y^k}{\partial x(u)^k}$ is the gradient of the dynamic model Y ; $x(u)^k$ and $x(u)^{k+1}$ are the values of the control signal in the k and $k+1$ iteration; and α is the size of the steps in the direction of minimization. The calculation of $\frac{\partial Y^k}{\partial x(u)^k}$ depends on availability of a differentiable model.

However, a comprehensive physical model of complex systems usually does not exist. In this case, using artificial neural networks in the controller is a good choice because ANNs are known for their function approximation capability. As introduced in section 2.4, properly designed and trained neural networks are capable of approximating any linear or nonlinear function to the desired degree of accuracy and they are noise tolerant.

Therefore, an ANN is used as the main part of the controller in the current work. The ANN is used for identification of the system, i.e., as a system model. The system model is trained off-line using the system inputs as inputs and the measured vibration responses of the plant at some point of interest as the output. The input signal of the plant is also the reference signal to the controller as shown in Figure 3-1.

Figure 3-2 shows the training process of the ANN model, where $x(p)$ represents the primary inputs to the plant, $x(u)$ represents the control inputs to the system and Y is the vibration response of the plant measured at some point of interest, which also is the expected output of the ANN model. The error, i.e., e , is the difference between the measured plant output values, i.e., Y , and predicted values of the ANN model, i.e., Y' . The error is backpropagated to train the ANN model. After training the ANN model, the model is used in the controller for AVC of the plant.

Figure 3-2: Training of the ANN model

To make the AVC system simple to implement, in the current work, the controller is a static ANN controller, which includes two main parts (see Figure 3-3). The first part is the trained ANN model. The second part is a minimization module which generates random control inputs $x(u)'$ to the ANN model and finds the minimum ANN model output value. Literally, the controller works in this way: for given primary inputs to the plant, i.e., $x(p)$, the ANN controller generates random control inputs $x(u)'$;

the combination of $x(p)$ and $x(u)'$ are sent to the ANN model as inputs; for each combination of $x(p)$ and $x(u)'$, an output, i.e., y' , is obtained and sent to the minimization module; the minimization module finds the minimum ANN model output, i.e., Y'_{min} , based on the given primary input to the plant and sends the corresponding signal, i.e., $x(u)'$, to the plant as control signal, i.e., $x(u)$.

Figure 3-3: Static ANN-based AVC methodology

The inputs of the ANN include the primary inputs to the plant, i.e., $x(p)$, and the control inputs to the system, i.e., $x(u)$. The output of the ANN model is the vibration responses of the plant measured at some point of interest, i.e., Y . The training data, i.e., $x(p)$, $x(u)$ and Y , are obtained by experiments. To get a robust neural network model, which means a model affected minimally by external sources of variability, the experiments need to be designed first. The design of experiments will be discussed in the experimental AVC design (see section 4.6.2).

The minimization module in the ANN controller uses simulated annealing and resilient propagation algorithms. They will be introduced in the following two sections.

3.2 Simulated Annealing

Simulated annealing is a technique that has attracted significant attention for optimization problems of large scale, especially ones where a desired global extremum is hidden among many local extrema. Simulated annealing is essentially an iterative random search procedure with adaptive moves along the coordinate directions [17]. It permits uphill moves under the control of a probabilistic criterion, thus tending to avoid the first local minimum encountered. Figure 3-4 represents the algorithm of this procedure [17].

It is common to combine simulated annealing with other minimization algorithms or training processes. In the current work, simulated annealing is utilized in the minimization module of the ANN plant model.

Figure 3-4: Simulated Annealing minimization algorithm [17]

3.3 Resilient Propagation

Resilient propagation (RPROP) performs a local adaptation of the weight updates according to the behavior of the error function [18]. Only the sign of the derivative is used to determine the direction of the weight update; the magnitude of the derivative has no effect on the weight update. The size of the weight change is determined by a separate update-value, i.e., Δ_{ij} . The update value for each weight and bias is increased by calculated factors. The adaptive updated -value, i.e., Δ_{ij} , evolves during the learning process based on its local sight on the error function E according to the following learning rule [18]:

$$\Delta_{ij} = \begin{cases} \eta^+ * \Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} * \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ \eta^- * \Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} * \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ \Delta_{ij}^{(t-1)}, & \text{else} \end{cases} \quad (3-1)$$

where $0 < \eta^- < 1 < \eta^+$, and w_{ij} is the weight from neuron j to neuron i .

Verbalized, the adaptation rule works as follows [18]: every time the partial derivative of the corresponding weight w_{ij} with respect to E changes its sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum, the update-value is decreased by the factor η^- . If the derivative retains its sign, the update-value is slightly increased by factor η^+ in order to accelerate convergence in shallow regions [18].

Once the update-value for each weight is adapted, the weight-update itself follows a very simple rule: if the derivative is positive (increasing error), the weight is decreased by its update-value; if the derivative is negative, the update-value is added

[18]:

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)}, & \text{if } \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ +\Delta_{ij}^{(t)}, & \text{if } \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ 0, & \text{else} \end{cases} \quad (3-2)$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \quad (3-3)$$

However, there is one exception: if the partial derivative changes sign, i.e., the previous step was too large and the minimum was missed, the previous weight-update is reverted [18]:

$$\Delta w_{ij}^{(t)} = -\Delta w_{ij}^{(t-1)}, \quad \text{if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} * \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \quad (3-4)$$

Due to that “backtracking” weight-step, the derivative is supposed to change its sign once again in the following step. In order to avoid a double punishment of the update value, there should be no adaptation of the update-value in the succeeding step.

In the current work, RPROP is used in the minimization module of the ANN control model.

3.4 ANN Model Design

In the proposed AVC system, the ANN model is used for function approximation (see section 2.4). The design of the ANN model in the ANN controller is based on the applied AVC strategy and the actual experimental setup. Generally, the design steps are as follows. First, training data for the ANN model are collected via experiments according to the AVC strategy presented in section 3.1 (see section 4.6.2 for actual examples). Then, the training data are analyzed in order to choose a normalization

method. The general network architecture of the ANN model is then designed and the suitable learning algorithm is chosen. Finally, the ANN model is trained properly to avoid overfitting. The network architecture may be modified in this step to get the better function approximation effect.

The actual design of the ANN model based on the experimental setup in the current work is presented in the following chapter.

CHAPTER 4

EXPERIMENTAL SETUP

The methodology presented in Chapter 3 is evaluated experimentally through active vibration control (AVC) of a cantilever beam. The experimental setup is described in this chapter.

4.1 Hardware Setup

Figure 4-1 shows the top view of the hardware setup developed for the experiments. The beam is made of plain carbon steel and has the dimensions: 550 mm x 25 mm x 4.5 mm. Two electromagnetic shakers are used to provide the primary disturbance force (shaker 1) and the control force (shaker 2) to the beam. These shakers are located at 145 mm and 373 mm from the clamped end, on each side of the beam, respectively. Since the shakers have significant mass, and mass loading will lower the apparent measured frequency ($\omega = \sqrt{\frac{k}{m}}$), to minimize the effect of the shakers on the structure, the shakers are attached to the beam through stingers. The stingers serve to isolate the shakers from the structure, reduce the added mass, and cause the force to be transmitted axially along the stingers. The primary shaker is attached to the beam firmly; but the control shaker simply pushes up against the beam. The resulting preload is used to maintain contact between the control shaker and the beam. The objective is to minimize the vibration of the beam at the proximity sensor location, which is 498 mm from the clamped end of the beam.

Figure 4-1: Hardware setup for implementing AVC design

The data acquisition (DAQ) board has four 12-bit analog inputs with a maximum sampling rate of 5 Mega-Sample (MS)/s. It also provides two 16-bit analog outputs with a maximum update rate of 2.5 MS/s (see Appendix A for the DAQ board specifications). Figure 4-2 is a sketch of the experimental hardware setup, which shows some functions of the DAQ board. Two analog outputs, i.e., “AO 1” and “AO 2”, and one analog input, i.e., “AI ”, are used in the experiments.

Figure 4-2: Schematic diagram of experimental setup

The vibration is measured using an inductive proximity sensor (see Appendix B for sensor specifications). The two electromagnetic shakers are driven by two 40W amplifiers. Figure 4-3 shows three photographs of the experimental setup as described.

(a) Experimental hardware setup

(b) Close-up view 1 of experimental hardware setup

Figure 4-3 (a)-(b): Photographs of the experimental hardware setup

(c) Close-up view 2 of experimental hardware setup

Figure 4-3 (c): Photographs of the experimental hardware setup

4.2 Software Environment

In the current work, the operation system is Microsoft Windows 2000. LabVIEW 7.1 and LabWindows/CVI 7.0 are used for measurement, signal processing and the development of user interfaces. MATLAB 7.0 is used for neural network design, implementation, visualization and simulation, as well as data analysis and graphing. Microsoft Visual C++ 6.0 is used for algorithm implementation (e.g., simulated annealing and resilient propagation) and for implementing neural networks in the form

of Dynamic Link Library (DLL) files. ANSYS 7.0 is used for the Finite Element Method (FEM) analysis of the beam vibration.

Figure 4-4 shows one example of user interfaces developed in the current work. In this user interface, the amplitudes, frequencies and phases of the primary and control signals can be input by typing or turning the small nubs. The calculated control signal parameters, measured analog input maximum peak-peak amplitudes, an artificial neural network (ANN) model outputs and the elapsed time of the system are displayed numerically. The control, primary and analog input signals are displayed graphically. And two FFT graphs show the phases and magnitudes of the analog input signal.

Figure 4-4: An example of user interfaces used in the current work

4.3 Signal Processing

4.3.1 Sampling Time

In the current work, sampling time is set to one second. Therefore, signal processing, such as Fast Fourier Transform (FFT), is based on one-second-period signals. As a result, for signals with frequencies of integer hertz, signal leakage is eliminated. For example, Figure 4-5 shows the FFT graphs of the analog input signal from the sensor when a sinusoidal signal (peak-to-peak 0.01V, 75Hz) is sent to the primary shaker by using the experimental setup shown in Figure 4-1. When the sampling time is one second, this is very little leakage, as shown in Figure 4-5 (a). When the sampling time is 0.1 second, leakage occurs, as shown in Figure 4-5 (b).

(a) Sampling time = 1s (no leakage)

(b) Sampling time = 0.1s (leakage occurs)

Figure 4-5: FFT graphs of a same signal with different sampling time

For the signals with non-integer frequencies, appropriate window functions are utilized to correct the leakage.

4.3.2 Window Functions

Leakage can be corrected to some degree by the use of some window functions. There are several different window functions. Each window function has its own strengths and weaknesses. For the sine wave or combination of sine wave signals, the Hanning window function is a good choice; while for a white noise signal, the Uniform

window function is an appropriate choice [35]. Therefore, in the current work, the Hanning window function is utilized for sinusoidal signals and the Uniform window function is utilized for white noise signals.

4.3.3 Sampling Rate

In the current work, the sampling rate is set to 1000 Hz. This is good for fast FFT calculation [35] and to avoiding aliasing. According to Shannon's sampling theorem, to avoid aliasing, the signal should be sampled at a rate at least two times of the highest frequency in the signal [1]. As will be discussed in the following chapter, the highest frequency in the signals of the current work is 120 Hz. Therefore, the sampling rate is more than 9 times of the highest frequency. In this case, aliasing should be avoided very well.

4.3.4 Magnitude Scales in FFT

Magnitude can be displayed on either a linear scale or a decibel (dB) scale in the FFT display [35]. The linear scale shows the amplitudes as they are, while the decibel scale is a transformation of the linear scale into a logarithmic scale. The decibel is the unit of a ratio. Equation (4-1) defines the decibel in terms of power and Equation (4-2) defines the decibel in terms of amplitude [35].

$$dB = 10 \log_{10} \frac{P}{P_r} \quad (4-1)$$

where P is the measured power, P_r is the reference power, and $\frac{P}{P_r}$ is the power ratio.

$$dB = 20 \log_{10} \frac{A}{A_r} \quad (4-2)$$

where A is the measured amplitude, A_r is the reference amplitude, and $\frac{A}{A_r}$ is the voltage ratio [35].

When the measured signal is very small compared with the reference, using a

linear scale is better than a decibel (dB) scale for clarity. Figure 4-6 shows the FFT graphs of the analog input signal from the sensor when a sinusoidal signal (peak-to-peak amplitude = 0.01V and frequency = 50 Hz) is sent to the control shaker by using the experimental setup. It is obvious that displaying magnitude on a linear scale is much clearer than on a decibel scale here. For clarity, most of the FFT magnitude scales in this report are on linear scales.

(a) Magnitude displayed on a linear scale

(b) Magnitude displayed on a decibel scale

Figure 4-6: FFT graphs of a same signal on a linear scale and a dB scale

4.4 Bending Vibration Analysis of the Beam

In the proposed experimental setup, the vibration of the cantilever beam in the direction perpendicular to its length is considered. Such vibrations are known as transverse or flexural vibrations.

The cantilever beam of this experiment can be considered an Euler-Bernoulli beam because its length/width ≥ 10 and length/thickness ≥ 10 [1]. Therefore, the effects of shear deformation and rotary inertia are ignored in the vibration analysis.

When the energy dissipation of the beam is taken into consideration, the damping ratios ζ_n (the n th modal damping ratio) are chosen based on experience or on

experimental measurements. Usually, ζ_n is a small positive number between 0 and 1, with most common values of $\zeta_n \leq 0.05$ [1] [2]. The damped natural frequency is:

$$\omega_{dn} = \omega_n \sqrt{1 - \zeta_n^2} \quad (4-3)$$

where ω_n is the undamped natural frequency.

If the energy dissipation of the beam (see Figure 4-7) is ignored, the natural frequency ω_n and the mode shape $X_n(x)$ of the beam can be calculated as [1] [2]:

Figure 4-7: A cantilever beam

$$\omega_n = \beta_n^2 \sqrt{EI / \rho A} \quad (4-4)$$

$$X_n(x) = \cosh \beta_n x - \cos \beta_n x - \sigma_n (\sinh \beta_n x - \sin \beta_n x) \quad (4-5)$$

$$\sigma_n = \frac{\sinh \beta_n l - \sin \beta_n l}{\cosh \beta_n l + \cos \beta_n l} \quad (4-6)$$

where $\beta_n l$ are called the weighted frequencies, l is the length of the beam, E is the Young's elastic modulus for the beam, I is the cross-sectional area moment of inertia, and ρ is the density of the beam.

From [1], the following parameters can be found:

$$\beta_1 l = 1.87510407; \quad \beta_2 l = 4.69409113; \quad \beta_3 l = 7.85475744; \dots$$

$$\sigma_1 = 0.7341; \quad \sigma_2 = 1.0185; \quad \sigma_3 = 0.9992; \dots$$

For the beam used in the current work,

$$l = 0.55\text{m};$$

$$E \approx 2 \times 10^{11} \text{ N/m}^2 ; [33]$$

$$I = (1/12)(2.5 \times 10^{-2})(4.5 \times 10^{-3})^3 \approx 1.9 \times 10^{-10} \text{ m}^4 ; [38]$$

$$\rho \approx 7.84 \times 10^3 \text{ Kg/m}^3 ; [34]$$

$$A = (2.5 \times 10^{-2})(4.5 \times 10^{-3}) = 1.125 \times 10^{-4} \text{ m}^2 .$$

So the first three natural frequencies can be calculated by equation 4-4 as:

$$\omega_1 \approx 76.7 \text{ rad/s}, \quad f_1 = \frac{\omega_1}{2\pi} \approx 12.20 \text{ Hz}$$

$$\omega_2 \approx 477.3 \text{ rad/s}, \quad f_2 = \frac{\omega_2}{2\pi} \approx 75.96 \text{ Hz}$$

$$\omega_3 \approx 1337.1 \text{ rad/s}, \quad f_3 = \frac{\omega_3}{2\pi} \approx 212.80 \text{ Hz}$$

The first three mode shapes are plotted in Figure 4-8:

(a) Mode 1

Figure 4-8 (a): The first three mode shapes of the cantilever beam

(b) Mode 2

(c) Mode 3

Figure 4-8 (b)-(c): The first three mode shapes of the cantilever beam

The above calculation results of the natural frequencies and the mode shapes of the beam have also been confirmed by using the software ANSYS via the Finite Element Method (FEM).

4.5 Natural Frequencies of the Overall System

The method used to measure the natural frequencies of the whole experimental system including the beam and two shakers is via sending uniform white noise signals to the shakers and analyzing the corresponding Fast Fourier Transform (FFT) responses of the system at the sensor location.

Uniform white noise generates a signal that contains a uniformly distributed, pseudorandom pattern whose values are in the range $[-a: a]$, where a is the absolute value of amplitude [35]. Ideal white noise has equal power per unit bandwidth, resulting in a flat power spectral density across the frequency range of interest. Figure 4-9 shows the FFT response of the system output when uniform white noise ($a = 0.01V$) is sent to the primary shaker alone; to the control shaker alone; and to both shakers respectively. There is almost no jitter for the spectra of (a); but there are many jitters for the spectra of (b) and some jitters for the spectra of (c). However, the graphs show that the first natural frequency is around 34 Hz and the second natural frequency is around 130 Hz. Because of spectral leakage, 34 Hz and 130 Hz are approximate values of the first and second natural frequencies (see section 4.3.1).

(a) Uniform white noise ($a = 0.01V$) is sent to the primary shaker only

Figure 4-9 (a): Measuring the system natural frequencies via experiments

(b) Uniform white noise ($a = 0.01V$) is sent to the control shaker only

(c) Uniform white noise ($a = 0.01V$) is sent to both shakers

Figure 4-9 (b)-(c): Measuring the system natural frequencies via experiments

The main reason for the complex behavior of the system is the control shaker (actuator). The control shaker makes the FFT spectra of the system output jitter, because it is not firmly attached to the beam. It simply pushes up against the beam (see section 4.1). When the amplitude of vibration of the beam is large, the control shaker may lose contact with the beam, and therefore makes the system more complicated by introducing nonlinearity.

4.6 Introducing Nonlinearity to the System

For the purpose of testing performance of the system for nonlinear control problems, nonlinearity is introduced into the experimental arrangement. This can be done in two different ways (see Figure 4-10) in the experimental setup. The first way is by not attaching the control source shaker to the beam, but simply pushing it up against the beam. The resulting preload is used to maintain contact between it and the beam. By increasing the driving force of the primary disturbance, the control shaker must also drive harder to suppress the primary disturbance, which in this case will cause the control shaker to rattle as it loses the contact with the beam, and therefore will make the resultant error signal spectrum “noisier”.

Figure 4-10: Two ways to introduce nonlinearity

The effect of the first way is shown by the two graphs in Figure 4-11, which are produced when a sinusoidal signal (50Hz) is sent to the control shaker only. When the control shaker is driven soft (0.005V), the resultant error signal spectrum is very clean,

as shown in Figure 4-11 (a). Only the 50Hz peak (magnitude > 0.046) is evident, and all the other peaks are too small (magnitudes < 0.002) to be compared with this one. The small 60Hz peak is due to electrical noise, the small 100Hz peak is the first harmonic of 50Hz signal and the small 120Hz peak is the first harmonic of the electrical noise. However, when the control shaker is driven harder (0.1V), the resultant error signal spectrum becomes “noisier” with the comparable harmonics of the signal, as shown in Figure 4-11 (b).

Figure 4-11: Spectrum comparison between driving the control shaker soft and hard

The second way for introducing nonlinearity is by bandpass filtering the analog input signal from the sensor to provide a slight bias to the higher frequency harmonics, e.g., 100 Hz, thus exaggerating the relative importance of the harmonics in the spectrum.

CHAPTER 5

EXPERIMENTAL IMPLEMENTATION

5.1 AVC System Design

Based on the methodology discussed in Chapter 3 and the experimental setup described in Chapter 4, an active vibration control (AVC) system was designed as shown in Figure 5-1.

The Actuator in Figure 5-1 is the control shaker in Figure 4-1. The artificial neural network (ANN) model has five inputs and one output. The five inputs are the amplitude of the voltage signal sent to the primary shaker (V1), the frequency of that signal (F1), the amplitude of the voltage signal sent to the control shaker (V2), the frequency of this signal (F2) and the phase difference between the signals of the two shakers (PD). V1 and F1 are $x(p)$ in Figure 5-1, and V2, F2 and PD are $x(u)$ in the same figure. The ANN output is the amplitude of the expected analog signal from the sensor (see Figure 4-1). The controller's task is to find a suitable control signal to drive the control shaker to suppress the plant vibration at the sensor location based on the reference signal. The reference signal is the driving signal to the primary shaker. As shown in Figure 5-1, the controller comprises the ANN model and a minimization module that uses simulated annealing (see section 3.2) and the resilient propagation algorithm (see section 3.3) to find the possible minimum value of the ANN outputs based on the given reference signal data, i.e., V1 and F1. Then the control signal data, i.e., V2, F2 and PD, which are the inputs to the ANN to get the possible output minimum value, are sent to the control shaker to suppress the plant vibration at the sensor location.

To use the resilient propagation algorithm, the ANN output Y' is considered as a function of the signal amplitude of the control shaker ($V2'$), the signal frequency of the

control shaker ($F2'$) and the phase difference between the signals of the two shakers (PD'), i.e.,

$$Y' = f(F2', V2', PD') \quad (5-1)$$

The resilient propagation algorithm is utilized to find the smallest possible output value Y , and the corresponding $F2$, $V2$ and PD (see section 3.3). Simulated annealing is used to help the resilient propagation algorithm to avoid being trapped by a local minimum (see section 3.2).

Figure 5-1: The proposed AVC system

5.2 ANN Training Data

As discussed in Chapter 3, after the control strategy is set up, the first step to

design the ANN model is to collect training data based on the control strategy and experimental setup. The training data should be collected via experiments based on the control strategies and the experimental setup. To get a robust training, which means a training affected minimally by external sources of variability, the experiments to collect training data need to be designed first.

Generally, experiments are used to study the performance of processes and systems. The process or system can be represented by the model shown in Figure 5-2. The process transfers some inputs into an output that has one or more observable responses [27]. Some of the process variables (factors) $x_1, x_2, \dots, x_p$ are controllable, whereas other variables $z_1, z_2, \dots, z_q$ are uncontrollable.

Figure 5-2: General model of a process or system [27]

The objectives of the experiment may include the following:

1. Determining which factors are most influential on the response y ;
2. Determining where to set the influential x 's so that the response y is almost always near the desired nominal value;
3. Determining where to set the influential x 's so that variability in y is small;
4. Determining where to set the influential x 's so that the effects of the uncontrollable variables are minimized.

The general approach to planning and conducting the experiment is called the strategy of experimentation. There are several strategies that an experimenter could use, such as best-guess approach, one-factor-at-a-time approach, factorial design and fractional factorial design.

Factorial design is an experimental strategy in which factors are varied together, in stead of one at a time. The fractional factorial design is a variation of basic factorial design in which only subsets of the runs are made and it considers the interaction among the factors. In this project, the fractional factorial design is used for the design of experiments to obtain the training data for the ANN model.

The objective of the experiments in the current work is to determine the influence of the controllable factors, i.e., x 's, to the response y and where to set some of the influential x 's so that the response y can be the smallest possible value (see Figure 5-2). The following shows how the fractional factorial experiment was used. This example shows the design of experiments to collect 50 sets of training data for the AVC system.

In this case, the output y is the amplitude of analog signal from the sensor. The five controllable factors, i.e., x 's, are considered as:

1. The phase difference between the signals of the two shakers (PD);
2. The signal amplitude of the primary shaker (V1);
3. The signal frequency of the primary shaker (F1);
4. The signal amplitude of the control shaker (V2);
5. The signal frequency of the control shaker (F2).

The ranges over which the factors will be varied are decided by the regions of interest for each variable and the hardware performance limitations. For example, since the frequency response ranges for the two amplifiers are 20 Hz ~ 30 KHz, and the vibration frequency range of interest is 30 Hz ~ 120 Hz in the current work, the frequency range 30 Hz ~ 120 Hz is used for the signals. To find out the proper amplitude ranges for the signals sent to the shakers, many experiments were undertaken.

Considering the measurement range of the sensor and the hardware setup, according to the results of these experiments, the peak-to-peak amplitude range for the signals sent to the shakers is set to 0.002 V ~ 0.033 V. The phase difference range can be set to 0 deg. ~ 360 deg.

As an example, if only 50 sets of data are desired, the experiments can be designed as Table 5-1 [28]. Five levels are used for the signal amplitudes and frequencies, while ten levels are used for the phase difference. The phase difference is considered at 0 degree, 36 degrees, 72 degrees, 108 degrees, 144 degrees, 180 degrees, 216 degrees, 252 degrees, 288 degrees and 324 degrees. The only response variable of interest, i.e. y , is the peak-to-peak amplitude of the sensor's output signal, which needs to be measured in these experiments. Table 5-1 shows the 50 sets of data for the experimental design of this example [28]:

Expt. No.	Phase Diff. (Degree)	V1 (Volt)	V2 (Volt)	F1 (Hz)	F2 (Hz)	Sensor Output (Volt)
1	0	0.002	0.002	30.000	30.000	
2	0	0.005	0.005	55.000	55.000	
3	0	0.010	0.010	78.000	78.000	
4	0	0.020	0.050	98.000	98.000	
5	0	0.033	0.100	120.000	120.000	
6	36	0.002	0.005	78.000	98.000	
7	36	0.005	0.010	98.000	120.000	
8	36	0.010	0.020	120.000	30.000	
9	36	0.020	0.033	30.000	55.000	
10	36	0.033	0.002	55.000	78.000	
11	72	0.002	0.010	120.000	55.000	
12	72	0.005	0.020	30.000	78.000	
13	72	0.010	0.033	55.000	98.000	
14	72	0.020	0.002	78.000	120.000	
15	72	0.033	0.005	98.000	30.000	
16	108	0.002	0.020	55.000	120.000	
17	108	0.005	0.033	78.000	30.000	
18	108	0.010	0.002	98.000	55.000	

19	108	0.020	0.005	120.000	78.000	
20	108	0.033	0.010	30.000	98.000	
21	144	0.002	0.033	98.000	78.000	
22	144	0.005	0.002	120.000	98.000	
23	144	0.010	0.005	30.000	120.000	
24	144	0.020	0.010	55.000	30.000	
25	144	0.033	0.020	78.000	55.000	
26	180	0.002	0.002	98.000	120.000	
27	180	0.005	0.005	120.000	30.000	
28	180	0.010	0.010	30.000	55.000	
29	180	0.020	0.050	55.000	78.000	
30	180	0.033	0.100	78.000	98.000	
31	216	0.002	0.005	30.000	78.000	
32	216	0.005	0.010	55.000	98.000	
33	216	0.010	0.020	78.000	120.000	
34	216	0.020	0.033	98.000	30.000	
35	216	0.033	0.002	120.000	55.000	
36	252	0.002	0.010	78.000	30.000	
37	252	0.005	0.020	98.000	55.000	
38	252	0.010	0.033	120.000	78.000	
39	252	0.020	0.002	30.000	98.000	
40	252	0.033	0.005	55.000	120.000	
41	288	0.002	0.020	120.000	98.000	
42	288	0.005	0.033	30.000	120.000	
43	288	0.010	0.002	55.000	30.000	
44	288	0.020	0.005	78.000	55.000	
45	288	0.033	0.010	98.000	78.000	
46	324	0.002	0.033	55.000	55.000	
47	324	0.005	0.002	78.000	78.000	
48	324	0.010	0.005	98.000	98.000	
49	324	0.020	0.010	120.000	120.000	
50	324	0.033	0.020	30.000	30.000	

Table 5-1: Experimental design example for collecting 50-set data

In the actual experiments, a total of 500 sets data are collected to train the ANN in the control system.

5.3 Normalization

The values of all the data must be normalized for efficient processing by the ANN (see section 2.4.5). In the current work, all the experimental input and output data are positive values. The data are normalized to a range of 0.1 to 0.9 by using the following equation:

$$x_{scaled} = 0.8 * \frac{x - x_{min}}{x_{max} - x_{min}} + 0.1 \quad (5-2)$$

where x is the real value, x_{scaled} is the normalized value, x_{min} is the minimum value and x_{max} is the maximum value of one input or output.

5.4 ANN Architecture

In this work, multilayer feedforward ANNs are utilized (see section 2.4.1). The ANN architecture used here has five inputs, one hidden layer of log-sigmoid neurons and one output layer of one log-sigmoid neuron (because there is only one output). The output layer uses a log-sigmoid transfer function because the output of the ANN needs to be constrained to a range of 0 to 1 as discussed in last section (see section 2.4.1).

To find the suitable number of neurons in the hidden layer, many experiments have been done to see how many neurons in the hidden layer can provide the smallest Mean Square Error (MSE) for validation data sets (see section 2.4). Early stopping and Bayesian regularization methods are used in MATLAB to improve generalization in these experiments (see section 2.4.6). Test data sets are also used to compare different models. The best ANN architecture found in the experiments for the AVC strategy is shown in Figure 5-3. The best number of neurons in the hidden layer is eight. The MSE

of validation data sets for this architecture is 0.168%.

Figure 5-3: ANN architecture of the proposed AVC system

5.5 Training Algorithms

All the training algorithms in MATLAB are utilized to train the ANN on trial. The apparent difference among the algorithms is the training speed. The Levenberg-Marquardt algorithm is the fastest one. However, the training speed is not very critical in the current work, because after the ANN is trained, it is then simply used as a part of the controller and is not modified in the control process. Moreover, when using early stopping to improve generalization, an algorithm that converges too rapidly should not be used [36].

In this case, the resilient backpropagation training algorithm is a good choice, because its training speed is not very fast, and, theoretically, the resilient backpropagation training algorithm can also help to reduce the harmful effects of using sigmoid transfer functions in the hidden layer and out put layer of the ANN model.

Multilayer networks typically use sigmoid transfer functions in the hidden layers.

These functions are often called "squashing" functions, since they compress an infinite input range into a finite output range [36]. Sigmoid functions are characterized by the fact that their slope must approach zero, as the input gets large. This causes a problem when using steepest descent to train a multilayer network with sigmoid functions, since the gradient can have a very small magnitude; and therefore, cause small changes in the weights and biases, even though the weights and biases are far from their optimal values [36]. In MATLAB, the resilient backpropagation training algorithm is used to eliminate these harmful "squashing" effects of the magnitudes of the partial derivatives.

Therefore, in the current work, the resilient backpropagation training algorithm is utilized to train the ANN model.

After the ANN model is trained, it works in the controller of the AVC system as shown in Figure 5-1. The work performance of this AVC system has been tested experimentally.

5.6 Results and Discussion

In order to test the performance of the proposed AVC system experimentally, a sinusoidal signal is sent to the primary shaker to generate the primary disturbance to excite the beam vibration. This signal is also sent to the ANN controller at the same time as a reference signal (see Figure 5-1). The controller generates another sinusoidal signal based on the reference signal, and sends the signal to the control shaker after six seconds to suppress the beam vibration at the sensor location as shown in Figure 4-1.

The frequency range for the signals is 30 Hz to 120 Hz and the amplitude range for the signals is 0.002 V to 0.033 V (see section 5.2). The first natural frequency of the system is around 34 Hz (see section 4.5).

All the analog input and analog output signals, and FFT (magnitude and phase)

are displayed on the user interface graphically (see Figure 4-4 for an example). The DAQ sampling rate is set to 1000 Hz to avoid aliasing (see section 4.3). All data are also recorded and saved as files on the computer hard disk for later analysis.

Figure 5-4 shows the first 10-second waveforms of the analog input signals from the sensor. These waveforms represent the beam vibration at the sensor location. Only the primary shaker is driven in the first six seconds, and then, the control shaker is also driven by a signal generated by the ANN controller. For example, in Figure 5-4 (a), in the first six seconds only a primary sinusoidal signal (peak-to-center amplitude $V1=0.03022$ V and frequency $F1=31.022$ Hz) was sent to the primary shaker. After six seconds, the control shaker is also driven by a sinusoidal signal generated by the ANN controller (peak-to-center amplitude $V2=0.033$ V, frequency $F2=45.2$ Hz and phase $PD=329.5$ degrees). The amplitude of the analog input from the sensor changes from 1.68 V to 2.295 V (AI: 1.68V $\rightarrow$ 2.295V) after the AVC.

- (a) $V1=0.03022$ V, $F1=44.75$ Hz; Control output: $V2=0.033$ V, $F2=45.2$ Hz, $PD=329.5$ deg.;
AI: 1.68V $\rightarrow$ 2.295V

Figure 5-4 (a): Performance of the proposed AVC system

- (b) $V1=0.0315\text{V}$, $F1=95.509\text{Hz}$; Control output: $V2=0.0231\text{V}$, $F2=118.57\text{Hz}$, $PD=230.1\text{ deg.}$;
AI: $0.159\text{V} \rightarrow 0.432\text{V}$

Figure 5-4 (b): Performance of the proposed AVC system

Generally, the experimental results of this AVC system are not satisfactory. In some cases the controller generated signals that made the beam vibration even larger, as shown in Figure 5-4. In other words, this AVC system has proved to be ineffective experimentally.

To find the reason, many experiments were conducted and the experimental data were analyzed. Experiment results show that the minimization module, which uses simulated annealing algorithm (see section 3.2) and the resilient propagation algorithm (see section 3.3) to find the minimum value (see Figure 5-1), worked very well in the controller; however, the ANN model did not.

The analysis shows that the ANN model is very sensitive to some of the inputs. For example, very small differences (smaller than the ANN training error) in the control signal frequencies (ANN inputs) can make big differences in the expected output from the sensor (ANN output). Yet in other cases the output of the network is insensitive to substantial changes in some inputs. It is believed that this is a consequence of squashing effect of normalization. For example, for some small outputs from the sensor, after data normalization, the differences between these outputs are “squashed” and therefore may be much smaller than the error of the ANN. The problem in this case cannot be solved

by just choosing other data normalization methods, and the resilient backpropagation training algorithm, which is supposed to be able to eliminate the harmful effects of using “squashing” functions (see section 5.5), can not work effectively here either. Therefore, the only way to make the AVC system work for the data ranges shown is to find a very good ANN model. This means, based on the experiment result analysis, the MSE of the ANN should be at least less than 10^{-7} ! To reach this MSE level, an impractical number of training data sets are needed because there are three variable inputs to the ANN (three dimensions). Finding such an ANN model is not practically possible.

The most probable reason why the proposed AVC system is not effective is that the function to be approximated by the ANN is very “unsmooth” in its variable space. The analysis of the experimental results has shown that very small differences in the control signal frequencies can make big differences in the expected analog outputs from the sensor, while in some cases, big control signal frequency differences can make very small differences in the expected outputs from the sensor. The complex behavior of the experimental system can also be shown in Figure 4-9. Figure 4-9 shows that the experimental system natural frequencies are non-stationary for different input combinations. Moreover, experiment results also show that the outputs from the sensor are very sensitive to the change of control signal frequencies, because very small differences (smaller than the ANN MSE) in the control signal frequencies can make big differences in the outputs from the sensor.

Since this AVC system has proved to be ineffective experimentally, it must be modified.

5.7 Modified AVC System

The above experiment results show that, in most situations, when the AVC

system works, the frequency of the control signal is almost the same as the frequency of the primary signal. Therefore, the first modification to the proposed AVC system is to set the frequency of the control signal the same as the frequency of the primary signal.

After this modification, the number of variable inputs to the ANN model is reduced to two, i.e. the amplitude of the control signal and the phase difference between the input signals of the two shakers. In this case, compared to equation 5-1, in the new AVC system,

$$Y' = f(V2', PD') \quad (5-3)$$

where Y' is the ANN output, $V2'$ is the input signal amplitude of the control shaker and PD' is the phase difference between the input signals of the two shakers. Because the variable dimensions reduced to two, the simulated annealing algorithm should be robust enough to work alone in the minimization module of the ANN controller. The simulated annealing algorithm works very reliably and provides high accuracy if it runs enough uniform random move cycles [17]. For example, if it runs one million uniform random move cycles, for two variable inputs (two dimensions) and a normalization range of 0.1 to 0.9, an accuracy of 0.001 for the normalized data can be reached. Therefore, the second modification to the proposed AVC system is using only simulated annealing in the minimization module of the controller.

The modified AVC system is shown in Figure 5-5.

Figure 5-5: The modified AVC system

Another modification to the old AVC system is the way the ANN model is trained. To train ANN model in the modified AVC system, the original frequency range, i.e. from 30 Hz to 120 Hz, is divided into five smaller sub-ranges, i.e., 30 Hz ~ 40 Hz, 40 Hz ~ 60 Hz, 60 Hz ~ 80 Hz, 80 Hz ~ 100 Hz and 100 Hz ~ 120 Hz. The ANN is trained with 100 training data sets for each sub-range separately.

The purpose of this modification is to reduce the harmful effects of normalization as discussed in last section. For example, because the first natural frequency of the system is around 34 Hz, when the signal amplitudes are the same, the signals in the sub-range 30 Hz ~ 40 Hz can generate much larger beam vibration than the signals in the sub-range 100 Hz ~ 120 Hz (see Figure 5-8 for some examples). Therefore, in the proposed AVC system, when all the training data are normalized for the whole range, i.e. 30 Hz to 120 Hz, after normalization, the output differences between the data in the

sub-range 100 Hz ~ 120 Hz are greatly squashed.

Since there are five smaller sub-ranges for the modified AVC system, after training, there are five weight files (one weight file for each sub-range respectively) for the ANN model. The ANN controller calls different weight file based on the frequency of the input signal to the primary shaker.

The design of experiments and normalization methods are similar to the methods used in designing the original AVC system.

Similarly, the ANN architecture used in the modified AVC system has four inputs, one hidden layer of log-sigmoid neurons and one output layer of one log-sigmoid neuron (because there is only one output). The output layer uses a log-sigmoid transfer function because the output of the ANN needs to be constrained to a range of 0 to 1 as discussed in section 2.4.1. The suitable number of neurons in the hidden layer is five for the ANN model in the modified system. The best ANN architecture found in the experiments for the modified AVC system is shown in Figure 5-6. The MSE of validation data sets for this architecture can get to 0.157%.

Figure 5-6: ANN architecture of the modified AVC system

Similarly, the resilient backpropagation training algorithm in MATLAB is utilized to train this ANN model. After the ANN model is trained, it works in the

controller of the modified AVC system as shown in Figure 5-5. The work performance of this modified AVC system has also been tested experimentally.

5.8 Results and Discussion of the Modified AVC System

In order to test the performance of the modified AVC system experimentally, similarly, a sinusoidal signal is sent to the primary shaker to generate the primary disturbance to excite the beam vibration. And this signal is also sent to the ANN controller at the same time as a reference signal (see Figure 5-5). The controller generates another sinusoidal signal, and sends the signal to the control shaker after six seconds to suppress the beam vibration at the sensor location as shown in Figure 4-1.

The frequency range for the signals is 30 Hz to 120 Hz and the amplitude range for the signals is 0.002 V to 0.033 V (see section 5.2). The first natural frequency of the system is around 34 Hz (see section 4.5).

All the analog input and analog output signals, and FFT (magnitude and phase) are displayed on all user interfaces graphically.

Figure 5-7 shows one control performance example of the modified AVC system. The sampling rate for data analysis in this current work is still 1000 Hz, although the sampling rate for drawing these figures is 500 Hz for clarity of presentation. Figure 5-7 (a) shows the user interface display. Figure 5-7 (b) shows the first 10-second waveform of the sensor output, which represents the beam vibration at the sensor location. Figure 5-7 (c) shows the vibration reduction on FFT graphs. In this experiment, only the primary shaker is driven in the first six seconds with a primary sinusoidal signal (peak-to-center amplitude $V_1=0.00318$ V and frequency $F=31.022$ Hz). After six seconds, the control shaker is also driven by a sinusoidal signal generated by the ANN controller (peak-to-center amplitude $V_2=0.033$ V, frequency $F=31.022$ Hz and phase

PD=0.0975 degrees). Figure 5-7 (b) shows that the amplitude of the signal from the sensor changes from 0.557 V to 0.146 V (AI: 0.557 V→0.146 V) after AVC, which represents a 73.8% reduction of vibration amplitude of the beam at the sensor location.

(a) The control performance user interface display

Figure 5-7 (a): Performance of the modified AVC system

(b) $V1=0.00318V$, $F=31.022Hz$; Control output: $V2=0.033V$, F , $PD=0.0975$ deg.;
AI: $0.557V \rightarrow 0.146V$

(c) Vibration reduction shown on FFT graphs

Figure 5-7 (b)-(c): Performance of the modified AVC system

Figure 5-8 shows more examples of the performance of the modified AVC system. The first 10-second waveforms of the sensor output, which represent the beam vibration at the sensor location, are plotted. For each test, only the primary shaker is driven in the first six seconds, and then, the control shaker is also driven by a signal generated by the ANN controller.

(a) $V1=0.00318V$, $F=31.022Hz$; Control output: $V2=0.033V$, F , $PD=0.0975$ deg.;
AI: $0.557V \rightarrow 0.146V$

(b) $V1=0.00309V$, $F=32.993Hz$; Control output: $V2=0.033V$, F , $PD=0.0325$ deg.;
AI: $1.789V \rightarrow 0.469V$

(c) $V1=0.0036V$, $F=33.19Hz$; Control output: $V2=0.033V$, F , $PD=0.0867$ deg.;
AI: $2.892V \rightarrow 0.470V$

Figure 5-8 (a)-(c): Performance examples of the modified AVC system

(d) $V_1=0.0035\text{V}$, $F=34\text{Hz}$; Control output: $V_2=0.033\text{V}$, F , $\text{PD}=0.1083\text{ deg.}$;

AI: $2.603\text{V} \rightarrow 1.085\text{V}$

(e) $V_1=0.0109\text{V}$, $F=34\text{Hz}$; Control output: $V_2=0.0328\text{V}$, F , $\text{PD}=0.0023\text{ deg.}$;

AI: $4.589\text{V} \rightarrow 2.781\text{V}$

(f) $V_1=0.0031\text{V}$, $F=35.11\text{Hz}$; Control output: $V_2=0.0329\text{V}$, F , $\text{PD}=0.2065\text{ deg.}$;

AI: $1.89\text{V} \rightarrow 0.889\text{V}$

Figure 5-8 (d)-(f): Performance examples of the modified AVC system

(g) $V1=0.0063V$, $F=35.11Hz$; Control output: $V2=0.0329V$, F , $PD=0.0545$ deg.;
AI: $3.181V \rightarrow 1.025V$

(h) $V1=0.0052V$, $F=55.921Hz$; Control output: $V2=0.0057V$, F , $PD=88.935$ deg.;
AI: $0.255V \rightarrow 0.078V$

(i) $V1=0.0328V$, $F=95.91Hz$; Control output: $V2=0.021V$, F , $PD=193.432$ deg.;
AI: $0.257V \rightarrow 0.078V$

Figure 5-8 (g)-(i): Performance examples of the modified AVC system

(j) $V1=0.0327V$, $F=110.012Hz$; Control output: $V2=0.031V$, F , $PD=211.709$ deg.;
AI: $0.279V \rightarrow 0.155V$

Figure 5-8 (j): Performance examples of the modified AVC system

The experimental results show that the modified AVC system works very effectively. In the examples in Figure 5-8, the vibration amplitudes are reduced by 39.4% (as in (e)) to more than 83.7% (as in (c)) for the given data range in the current work. In some cases, the amplitudes of the control signal reach the maximum value of its range, i.e., $0.033V$, which means the best control signal amplitudes may be out of the data range used in the current work. Several graphs, i.e., Figure 5-8 (a) to (g), show the control performance of frequencies around $34Hz$, because $34Hz$ is the first system natural frequency included in the data range. In all the tests, the ANN controller never generated a control signal to make the beam vibration larger. In many tests, the ANN controller of the modified AVC system can reduce the root mean square (RMS) vibrations by 10% to 90%. The reductions in the RMS vibrations have a very significant effect on the fatigue life of a structure in practical application. Generally, reducing the RMS vibrations by just 10% doubles the fatigue life [26]. Therefore the modified AVC system has proved to be very effective in the current work.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Contributions

A novel artificial neural network (ANN)-based active vibration control (AVC) technique has been developed. The proposed methodology was shown to be adaptable, simple to be implemented, and not affected by time delays in the system.

6.2 Conclusions

An AVC experimental system, including hardware setup and software development environment, has been successfully implemented. A static ANN is trained to model the plant using a backpropagation algorithm. The ANN model is obtained by the batch training mechanism that uses experimental input and output data. After training, the ANN is used as a part of a feedforward controller. The efficiency of this controller is shown through experimental tests.

6.3 Future Work

Modeling a system with a static neural network may not be robust enough for some applications. Therefore, as part of the future work, the system could be modeled as a dynamic neural network. Both off-line-training dynamic ANN-based AVC and on-line-training dynamic ANN-based AVC (as discussed in section 2.5) could be

researched and implemented. For off-line-training, the research should focus on the control strategy, while for on-line-training, it is important to find an effective training algorithm. To implement such a dynamic ANN-based AVC, a real-time experimental system is needed.

APPENDICES

Appendix A: DAQ Board Specifications

Appendix B: Proximity Sensor Specifications

S Series Multifunction DAQ Specifications

Specifications – PCI-6110, PCI-6111

These specifications are typical for 25 °C unless otherwise noted.

Analog Input

Accuracy specifications See Table 2, page 199

Input Characteristics

Number of pseudodifferential inputs:

PCI-6110 4

PCI-6111 2

Resolution 12 bits, 1 in 4,096

Maximum Sampling Rate

6110/6111 5 MS/s

Minimum sampling rate 1 kS/s

Streaming-to-disk rate 6110/6111 8 MS/s, system dependent

Input signal ranges

Input Range (Software, Selectable per Channel)	Bandwidth (MHz)
±42 V	5.5
±20 V	4.4
±10 V	7.2
±5 V	4.8
±2 V	4.8
±1 V	4.4
±500 mV	4.4
±200 mV	4.1

Input coupling DC or AC (software selectable)

Maximum working voltage for all analog input channels

Input Channels	Range	Maximum Working Voltage (Signal + Common Mode)
ACH<0..3>+	200 mV to 10 V 20 to 50 V	Should remain within ±11 V of ground Should remain within ±42 V of ground
ACH<0..3>-	All	Should remain within ±11 V of ground

Overvoltage protection

On/Off ±50 V

Inputs protected ACH<0..3>+, ACH<0..3>-

FIFO Buffer Size

6110/6111 8,192 Samples

Data transfers DMA, interrupts, programmed I/O

DMA modes Scatter-gather (single transfer, demand transfer)

Transfer Characteristics

Relative accuracy ±0.5 LSB typical, ±1 LSB maximum

DNL ±0.3 LSB typical, ±0.75 LSB maximum

No missing codes 12 bits, guaranteed

Spurious free dynamic range (SFDR), DC to 100 kHz

Range	SFDR (dB)
200 mV to 10 V	75
20 to 42 V	70

Effective number of bits (ENOB) 11.0 bits, DC to 100 kHz

Amplifier Characteristics

Input impedance

ACH<0..3>+ to ACH<0..3>-

Normal powered on 1 MΩ in parallel with 100 pF

Powered off 1 MΩ minimum

Overload 1 MΩ

Impedance to ground

ACH<0..3>- to ground 10 nF

Input bias current ±200 pA

Input offset current ±100 pA

CMRR, DC to 80 Hz

Range	CMRR (dB)
200 mV	72
500 mV	70
1 V	67
2 V	62
5 V	56
10 V	50
20 to 42 V	35

Dynamic Characteristics

Interchannel skew 1 ns typical; f_{in} = 100 kHz; 10 V range

System noise (LSB_{rms}, not including quantization)

Range	Noise
200 mV	1.0
500 mV	0.6
1 to 50 V	.05

Crosstalk -80 dB, DC to 100 kHz

Analog Output

Output Characteristics

Number of channels 2 voltage outputs

Resolution

6110/6111 16 bits, 1 in 65,536

Maximum update rate

1 channel 4 MS/s, system dependent

2 channel 2.5 MS/s, system dependent

Type of DAC Double buffered, multiplying

FIFO Buffer Size

6110/6111 2,048 Samples

Data transfers DMA, interrupts, programmed I/O

DMA modes Scatter-gather (single transfer, demand transfer)

Transfer Characteristics

Relative accuracy ±4 LSB typical, ±8 LSB maximum

DNL ±2 LSB typical, ±8 LSB maximum

Voltage Output

Ranges ±10 V

Output coupling DC

Output impedance 50 Ω ±5%

Short circuit current ±27 mA typical

Output stability Any passive load

Protection Short-circuit to ground

Power-on state 0 V, ±400 mV

Dynamic Characteristics

Settling time and slew rate

Settling Time for Full-Scale Step	Slew Rate
300 ns to ±0.01%	300 V/μs

Noise 1 mV_{rms}, DC to 5 MHz

Glitch energy (at mid-scale transition)

Magnitude 350 pV

Digital I/O

Number of channels 8 input/output

Compatibility 5 V TTL/CMOS

Power-on state Input (high impedance)

Data Transfers

6110/6111 Programmed I/O

S Series Multifunction DAQ Specifications

Specifications – PCI-6110, PCI-6111 (continued)

Digital logic levels

Level	Minimum (V)	Maximum (V)
Input low voltage	0	0.8
Input high voltage	2.0	5.0
Output low voltage (I _{out} = 24 mA)	–	0.4
Output high voltage (I _{out} = 13 mA)	4.35	–

Timing I/O

General-Purpose Up/Down Counter/Timers

Number of channels	2
Resolution	24 bits
Compatibility	5 V TTL/CMOS

Digital logic levels

Level	Minimum (V)	Maximum (V)
Input low voltage	0	0.8
Input high voltage	2.0	5.0
Output low voltage (I _{out} = 5 mA)	–	0.4
Output high voltage (I _{out} = 3.5 mA)	4.35	–

Base clocks available	20 MHz and 100 kHz
Base clock accuracy	±0.01%
Maximum source frequency	20 MHz
External source selections	PFI <0..9>, RTSI <0..6>, analog trigger; software selectable
External gate selections	PFI <0..9>, RTSI <0..6>, analog trigger; software selectable
Minimum source pulse duration	10 ns, edge-detect mode
Minimum gate pulse duration	10 ns, edge-detect mode
Data transfers	DMA, interrupts, programmed I/O
DMA modes	Scatter-gather (single transfer, demand transfer)

Frequency Scaler

Number of channels	1
Resolution	4 bits
Compatibility	5 V/TTL

Digital logic levels

Level	Minimum (V)	Maximum (V)
Input low voltage	0	0.8
Input high voltage	2.0	5.0
Output low voltage (I _{out} = 5 mA)	–	0.4
Output high voltage (I _{out} = 3.5 mA)	4.35	–

Base clocks available	10 MHz, 100 kHz
Base clock accuracy	±0.01%
Data transfers	Programmed I/O

Triggers

Analog Triggers

Number of triggers	1
Purpose	
Analog input	Start and stop trigger, gate, clock
Analog output	Start trigger, gate, clock
General-purpose counter/timers	Source, gate

Source

PCI-6110	ACH<0..3>, PFI0/TRIG1
PCI-6111	ACH<0..1>, PFI0/TRIG1

Level

Internal source, ACH<0..3>	±Full-scale
External source, PFI0/TRIG1	±10 V
Slope	Positive or negative; software-selectable
Resolution	8 bits, 1 in 256
Hysteresis	Programmable Bandwidth (-3 dB)
Internal source, ACH<0..3>	5 MHz
External source, PFI0/TRIG1	5 MHz

Digital Triggers

Number of triggers	2
Purpose	
Analog input	Start and stop trigger, gate, clock
Analog output	Start trigger, gate, clock
General-purpose counter/timers	Source, gate
Source	PFI<0..9>, RTSI<0..6>
Slope	Positive or negative; software selectable
Compatibility	5 V TTL
Response	Rising or falling edge
Pulse width	10 ns minimum

External Input for digital or analog trigger (PFI0/TRIG1)

Impedance	10 kΩ
Coupling	DC or AC
Protection	
Digital trigger	-0.5 to V _{CC} + 0.5 V
Analog trigger	
On/off/disabled	±35 V

Calibration

Recommended warm-up time	15 minutes
Calibration interval	1 year
Onboard calibration reference	
DC Level	5.000 V (±3.5 mV); actual value stored in EEPROM
Temperature coefficient	±0.8 ppm/°C maximum
Long-term stability	±6 ppm/√1,000h

RTSI

Trigger lines	7
---------------	---

Bus Interface

PCI	Master, slave
-----	---------------

Power Requirements

Device	+5 VDC (±5%)	Power Available at I/O Connector
PCI-6110	3.0 A	+4.65 to 5.25 VDC, 1 A
PCI-6111	2.5 A	+4.65 to 5.25 VDC, 1 A

Physical

Dimensions

Not including connectors	31.2 by 10.6 cm (12.3 by 4.2 in.)
I/O connector	68-pin male SCSI type

Environment

Operating temperature	0 to 45 °C
Storage temperature	-20 to 70 °C
Relative humidity	10 to 90%, noncondensing

Certifications and Compliances

CE Mark Compliance

*Bandwidth specifications are for signals on the (+) input with the (-) input at DC ground. The (-) input is slew rate limited to 24 V/μsec and has an additional 10 nF capacitance to ground.

AE SERIES ANALOG INDUCTIVE PROXIMITY SENSORS

M8 (8mm) metal – analog output

- 4 models available
- Compact metal housing
- Axial cable or M8 quick-disconnect models
- IP67 rated

AE Series M8 Analog Inductive Prox Selection Chart

Part Number	Price	Sensing Range	Housing	Output	Connection	Dimensions
AE9-10-1A	check	0 to 4.0mm (0-0.157in)	Shielded	0-10VDC	2m (6.5') axial cable	Figure 1
AE9-10-1F	check				M8 (8mm) connector	Figure 2

Specifications

	AE9-10-1
Output Type	0-10VDC
Resolution	≤1μm
Repeat Accuracy	±0.01mm
Material Correction Factors	See Proximity Sensor Terminology
Operating Voltage	15-30VDC
Ripple	≤20%
No-load Supply Current	≤10mA
Voltage Output Minimum Load	1kΩ
Voltage Drop	≤2.0 V
Time Delay Before Availability	≤50ms
Input Voltage Transient Protection	Up to 30VDC
Input Power Polarity Reversal Protection	Yes
Output Power Short-Circuit Protection	Yes (switch autoresets after overload is removed)
Temperature Range	-25° to +70° C (-13° to 158° F)
Temperature Drift	≤10% Sr
Protection Degree (DIN 40050)	IEC IP67
Housing Material	Chrome-plated brass
Sensing Face Material	PBT
Tightening Torque	4Nm (0.71lb./in.)
Weight (cable/M8 connector)	50g (1.76 oz.) / 20g (0.71 oz.)

Dimensions

Figure 1

Figure 2

Wiring diagrams

Connector

REFERENCES

1. D. Inman, "Engineering Vibration," Prentice-Hall, Englewood Cliffs, NJ, 1994.
2. W. Thomson, M. D. Dahleh, "Theory of Vibration with Applications," 5th ed., Prentice Hall, 1998.
3. C. R. Fuller, S. J. Elliott, and P. A. Nelson, "Active Control of Vibration," Academic, New York, 1995.
4. S. J. Elliott and P. A. Nelson, "Active noise control," IEEE Signal Processing Mag., vol. 10, pp. 12–35, 1993.
5. S. J. Elliott, "Signal Processing for Active control," Academic Press, 2001.
6. C. H. Hansen and S. D. Snyder, "Active Control of Noise and Vibration," E&FN Spon, London, U.K., 1997.
7. S. D. Snyder and N. Tanaka, "Active Control of Vibration Using a Neural Network," IEEE Transactions on Neural Networks, Vol. 6, No. 4, pp. 819-828, July 1995.
8. B. Widrow and S. D. Stearns, "Adaptive Signal Processing," Prentice-Hall, Englewood Cliffs, NJ, 1985.
9. J. J. Shynk, "Adaptive IIR Filtering," IEEE Acoust. Speech Signal Processing Mag., pp. 4–21, 1989.

10. L. Hakansson, "The Filtered-x LMS Algorithm," University of Karlskrona/Ronneby, Sweden.
11. S. Haykin, "Adaptive Filter Theory," 3rd ed., Prentice-Hall, Englewood Cliffs, NJ, 1996.
12. S. Haykin, "Neural Networks: A Comprehensive Foundation," 2nd ed., Prentice-Hall, Englewood Cliffs, NJ, 1999.
13. M. Hagan and H. Demuth, "Neural Network Design," PWS Publishing Co., Boston, MA, 1996.
14. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, "Numerical Recipes in C- The Art of Scientific Computing," Cambridge University Press, 1992.
15. S. M. Kuo and D. R. Morgan, "Active Noise Control Systems: Algorithms and DSP Implementations," Wiley, New York, 1996.
16. K.S. Narendra and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks," IEEE Trans, Neural Networks, Vol. 2, pp. 252-262, 1991.
17. A. Corana, M. Marchesi, C. Martini and S. Ridella, "Minimizing Multimodal Functions of Continuous Variables with the 'Simulated Annealing' Algorithm," ACM. Transactions on Mathematical Software, Vol. 13, No.3, pp. 262-280, September 1987.

18. M. Riedmiller, and H. Braun, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," Proceedings of the IEEE International Conference on Neural Networks, San Francisco, 1993.
19. M. Bouchard, B. Paillard, and C. T. LeDinh, "Improved Training of Neural Networks for the Nonlinear Active Control of Sound and Vibration," IEEE Transactions on Neural Networks, Vol. 10, No. 2, pp. 391-401, March 1999.
20. M. Bouchard, "New Recursive-Least-Squares Algorithms for Nonlinear Active Control of Sound and Vibration Using Neural Networks," IEEE Transactions on Neural Networks, Vol. 12, No. 1, pp. 135-147, January 2001.
21. T. Insperger, G. Stepan, P.V. Bayly, and B.P. Mann, "Experimental Investigation of Active Vibration Control Using Neural Networks and Piezoelectric Actuators," Institute of Physics Publishing, February 2001.
22. W. C. Choi and N. W. Kim, "Experimental Study on Active Vibration Control of a Flexible Cantilever Using an Artificial Neural-network State Predictor," Multimedia Institute, LG Electronics Ltd, 16 Umyon-dong, Sucho-go, 137-240, August 1996.
23. G. C. M. Deabreu, R. L. Teixeira and J. F. Ribeiro, "A Neural Network-Based Direct Inverse Control for Active Control of Vibrations of Mechanical Systems," SBRN, Vol. 00, No. 00, pp. 107-116, VI, 2000.
24. R. Jha and J. Rower, "Experimental investigation of active vibration control using neural networks and piezoelectric actuators," Institute of Physics Publishing, November 2002.

25. K. E. Rouch, S. G. Tewani, B. L. Walcott, T. R. Massa, R. W. Stephenson and L. S. Stephens, "Active vibration control device," U.S. Patent No. 5.170.103, 1992.
26. L E Pado and P F Lichtenwalner, "Neural Predictive Control for Active Buffet Alleviation," Proc. 40th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference and Exhibit (AIAA 99-1319, St. Louis, MO), pp. 1043–53, 1999.
27. D. C. Montgomery, "Design and Analysis of Experiments," 5th ed., John Wiley & Sons, 2001.
28. M. S. Phadke, "Quality Engineering Using Robust Design," Prentice-Hall, Englewood Cliffs, NJ, 1989.
29. U. Naterwalla, "Chatter-Free Milling And Optimized Material Removal Rates," <http://www.mmsonline.com/articles/0300sup.html>
30. <http://www.micromega-dynamics.com/vibration-strategies.htm>
31. <http://soundlab.cs.princeton.edu/learning/tutorials/DSP/DSP.html>
32. <http://www.dspguru.com/info/faqs/fir/basics.htm>
33. <http://www.engineeringtoolbox.com/>
34. http://www.mcelwee.net/html/densities_of_various_materials.html

- 35. <http://ni.com/>
- 36. <http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/>
- 37. <http://msdn.microsoft.com/library/>
- 38. http://darkwing.uoregon.edu/~struct/courseware/461/461_lectures/461_lecture28/461_lecture28.html