THE EVOLUTION OF 'BOXES' TO QUANTIZED INDUCTIVE LEARNING:

A STUDY IN INDUCTIVE LEARNING

BY

JAMES PASCOE

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Mechanical Engineering

APPROVED:

H. H. Robertshaw, Chairman

A. L. Wicks

H. H. Cudney

July 17, 1996
Blacksburg, Virginia

KEYWORDS: ARTIFICIAL INTELLIGENCE, MACHINE LEARNING, DAMAGE CONTROL

THE EVOLUTION OF 'BOXES' TO QUANTIZED INDUCTIVE LEARNING: A STUDY IN INDUCTIVE LEARNING

by

James W. Pascoe

Dr. Harry Robertshaw, Chairman

Department of Mechanical Engineering

(ABSTRACT)

An inductive learning method is analyzed for use in on-line control. The controller has the benefit of being designed without a system model and is able to adapt itself to varying system parameters. Numerical experiments were performed with the Quantized Inductive Learning (QIL) algorithm, an extension of 'Boxes', on simple linear systems and a simulated simply supported aluminum beam.

Concurrent with the simulations, a theoretical analysis of the learning mechanism was generated. The evaluation of several issues with the algorithm (performance indices, sampling periods, and level of quantizations) were studied to validate the theory. A comparison with state feedback was used to compare the effectiveness of this method with traditional model-based approaches. The results indicate the method learns a control function which moves the system from an arbitrary initial condition to equilibrium or rejects a sinusoidal disturbance. In both cases, the control is learned in absence of an *a priori* system model.

ACKNOWLEDGMENTS

This work involved several people who helped guide, support, and advise me and I would to take this opportunity to thank them. First, I would like to recognize the efforts of my thesis advisor, Dr. Harry Robertshaw. His advice and insights helped guide my work and provided the opportunity for this work . I would also like to thank the members of my committee, Dr. Harley Cudney and Dr. Alfred Wicks. Their instruction allowed me to understand this work more completely. Also, I appreciate the financial assistance provided by the Army Research Office's University Research Initiative Center Program, Grant No. 03-92-6-0181; Dr. Gary Anderson, Program Manager.

I would also like to thank several people in the lab who took the opportunity to discuss the research and offer some insight. They are as follows: Dr. Will Saunders, Dan Cole, Gary Fagan, Pete Tappert, and Anton Sumali. Additionally, several people helped by providing moral support and I would like to thank: David Coe, Francisco Rivera, Colin Heichman, and Jim LaPean.

Finally, I would like to thank my wife and parents without who this work would have not been completed. Their constant urging when I wanted to give up helped me finish this work and reach my own personal goals.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

CHAPTER 1

INTRODUCTION

Scientists postulated the idea of Artificial Intelligence (AI) even before the development of the computer. For decades, research towards the goal of a 'thinking machine' has been pursued. Until recently, the limiting factors in AI research were computing power and cost, but the advent of the personal computer has reduced the barrier of cost and increased available computing power. The result is many problems and processes have been solved or improved using the precepts and guidelines of AI research and associated algorithms.

There are many different tactics in applying AI. Historically, this has resulted in many different disciplines of science and engineering creating computer codes that solve problems specific to their field. Now, the AI community is trying to disseminate and generalize the knowledge for everyone's use. Consequently, the AI community can be organized into four broad areas. First, expert-based systems use databases of knowledge to seek the solutions of problems. By adapting the set of knowledge, expert systems are able to generate new rules for the problem at hand. The heuristics set by the programmer determine the effectiveness of this technique. The second area, classified as Neural Networks, model the processes of the human brain. The network consists of at least three layers. One layer is an input layer using sensor information. The last layer is an output layer generating the response of the network, and one or more hidden layers are used to process the relation between the input and output. The

network is then trained using positive and negative examples. Fuzzy logic represents the third broad area of AI. Fuzzy logic controllers can best be thought of as introducing the concept of grey to the binary black and white world of computers. Essentially, fuzzy logic sets adapt their programmed rules to achieve a best fit by modifying weights in a learning matrix. Finally, the category of Machine Learning encompasses other algorithms. The basic premise for this category is that the machine learns from the world around it. The programmer sets up the algorithm and then lets it run and learn by processing information given it. This area has the most flexible algorithms because the concepts for design are so varied. There are inductive and deductive techniques. Designers can also choose generalized code versus specialized code or abstract processing versus concrete processing. The programmer chooses based on the desired goal for the system. Genetic Algorithms are an example in this area using parallel processing concepts to model biological genetics in seeking the optimal answer for a problem. Recently, the area of controls engineering has seen the emergence of AI techniques from these four groups because of the various advantages each one offers in solving problems or improving performance of current techniques.

This thesis will analyze applying a machine learning technique known as Quantized Inductive Learning (QIL) for Active Damage Control (ADC). Active Damage Control is a process where a controller controls a system with damage. The types of problems ADC addresses are delamination control, fatigue damage, impact damage, and high dynamic or transient stresses in a system. Consequently, ADC is logically split into the two areas of controlling the system before and after damage occurs and preventing the growth of damage in the system. For a structural process, this would be accomplished by alleviating the high strains at a damage site.

The analysis of QIL begins in chapter two with a literature review on other traditional methods used for Active Damage Control, the history of QIL, and a brief review of other AI techniques used in closely associated problems. Chapter three presents the major issues associated with using the algorithm. Chapter four consists of a theoretical analysis of the QIL algorithm. Chapter five analyzes numerical examples and computer experiments performed with the QIL algorithm to answer the questions raised by the research issues. Finally, chapter six draws conclusions from the work, presents the contributions made to the field, and presents recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

The area of ADC is a rapidly expanding field of research because the advances in computer hardware have made its pursuit feasible. As previously mentioned, ADC can be viewed as two separate areas of prevention of damage growth and control of a system before and after damage occurs. Both areas potentially use damage identification in achieving their goals. This literature review begins then with a survey of some traditional methods and AI methods used to locate damage.

The research in damage identification has admittedly little experimental work to verify the theory and numerical examples. Most of these methods involve using a modal analysis of the system to detect damage. Some experiments on clamped-free beams were carried out by Ju and Mimovitich (1988). Their study was concerned with the dynamic behavior of cracked beams in relation to their acoustic emissions. The actual study of damage location was not a primary concern, but its effects were. Similar work was performed by Crawley and Ray (1988) on cracked free-free beams. Again location of the damage was not determined.

In 1990, the use of modal analysis for crack identification was analyzed for free-free steel beams (Gomes and Silva, 1990). The authors theoretically and experimentally verified that modal analysis could be used to detect, locate, and quantify the amount of damage to a beam. Their work consisted of artificial cracks (slots) as well as real line cracks on specimens. The real cracks were caused by fatiguing specimens until cracks

appeared. Their results show that modal analysis could be used. The method, however, had trouble locating cracks on nodes of mode shapes (specifically 2nd and 4th modes). In addition, small depth cracks (crack depth/beam thickness ratio < 1/8) had little influence on the natural frequencies of the beam and could not be detected. Thus, the method was effective, but had some deficiencies.

Glass and Macalou attempted to conduct system identification of a rotating truss system with simulated damage using an artificial intelligence search routine (Glass and Macalou, 1991). Their idea was that an AI algorithm would analyze the output from a smart structure and compare the results to internal models. The current model of the structure was then identified. The algorithm had the ability to learn new models and incorporate the new model into the search tree. A frequency domain metric based on the squared frequency error and the modal correlation coefficient was used for the analysis. The results indicate the algorithm successfully identified the various models and even created a new one. It should be noted that computer simulations of the truss were used with damage simulated by noise on the sensors.

The area of active damage control has been explored from a variety of sub-problems. Active Fatigue Control, Active Delamination Control, Active Vibration Control, and Active Shape Control are all instances leading to the goal of active damage control. These sub-problems lend themselves to a variety of methods for solutions. This review will focus on the traditional methods used for control.

Active delamination control of composite beams was examined by Hanagud, *et al.* (1989). His study used modal analysis to detect damage to a beam and a fixed gain

optimal controller to try and control the growth of the damage using piezoelectric sensors and actuators. He proposed that delamination growth was triggered by the axial stress loading of the delamination caused by cyclical transverse loading of the beam. The control of axial stress reduced strain energy release rates and hence reduced growth rate. The results show the retardation of growth in a graphite epoxy cantilevered beam.

In 1991, Shape Memory Alloys (SMAs) were used to reduce stress at a crack tip and increase the fatigue life of a structure (Rogers, Liang, and Li, 1991). The use of induced strain actuators in a cantilevered beam were studied. They proposed that, by reducing the stress and consequently strain in a beam, fatigue life around a crack would be increased. The results indicate several orders of magnitude increase in the fatigue life criterion over uncontrolled structures.

Active Vibration Control is an area which has received much attention in modern controls. The area solves many different classes of problems from disturbance rejection to vibration suppression. The disturbance compensation problem is a class of problems which can be directly applied to Active Damage Control. So far, many proposed solutions for Active Damage Control involve controlling the system to prevent damage growth in response to a structural disturbance.

Sievers and von Flotow published a concise review of traditional modern control techniques for narrow-band disturbance rejection. Their paper analyzed and compared five methods of disturbance rejection and extended some of these methods to a larger class of problems. Their main extension involved the LMS algorithm. One thing to

note is that all the techniques presented require a fairly accurate model of the plant and disturbance dynamics to achieve compensation of the disturbance.

Other work that has involved disturbance compensation is wave absorbing controllers (von Flotow and Schafer, 1986). The idea behind these controllers is to develop a detailed structural model of the system as a series of traveling waves. This type of model immediately lends itself to a controller design that can reflect and absorb these waves. The analysis is viewed as a velocity feedback controller with frequency dependent gains. It is a local method of control that affects the global structure. The primary design criterion for the controller is a condition of matched impedance with the system. Experimental work consisted of a 2.9m cantilevered steel beam. The theoretical simulations showed good control, but the real system was destabilized by the controller in the 15th mode of the beam. This destabilization, however, was attributed to hardware limitations and not the theoretical design of the wave absorbing controller.

In reaction to the detailed models required above, control engineers have investigated more robust methods of disturbance compensation. Positive Position Feedback (PPF) was investigated in its ability for active damage control systems (Fagan, 1994). Fagan extended the analog design methods of Caughey and Goh to a digital implementation. PPF is a collocated method of local control that adds damping to a system via a set of 'tuning' filters. The advantage of PPF is guaranteed global stability for a gain criterion. In addition, more than one filter can be tuned to an actuator allowing local multimodal control. The thesis examined a simply supported aluminum beam with one collocated

sensor-actuator pair. Three modes of vibration were successfully suppressed by the one actuator for a broad band disturbance.

Since 1980, Artificial Intelligence has been investigated as a means to extend the range of robust controllers to rely very little on plant models for design. The goal for these model-independent controllers is to use only sensor information to develop the appropriate control laws for the system. The general precept behind these controllers is to have an iterative scheme that adapts itself to the system. As more information is gained, the controller then becomes more accurate.

In addressing the AI controllers, some definition of concepts used in AI are presented. Kubat (1992) wrote an extensive tutorial on the idea of Machine Learning. The goal for him was to provide a basic outline of the types of algorithms classified as machine learning for use by many different disciplines. He defined machine learning as intelligent processing of incoming information with respect to future utilization. The utilization of this information is directed by the programmer for the algorithm. Kubat used several concepts to show the difference in using Induction vs. Deduction, Generalization vs. Specialization, Abstraction vs. Concretion, and Consistent vs. Complete knowledge. For him, Artificial Intelligence works with data and 'knowledge'. This 'knowledge' is generally given as heuristics or rules of thumb by the programmer. The interaction of data and knowledge provides the core of an AI algorithm to try and achieve the three goals of:

1. Ability to think and learn
2. Ability to interact intelligently
3. Ability to learn

With this brief background, the literature on Machine Learning in controls provides several examples of learning algorithms.

In 1980, Astrom developed self-tuning regulators (Astrom, 1980). The concept for these controllers is a transfer function of the system could be calculated at the k+1th time step from the k+1th state. The transfer function was formed by using a recursive least squares fit of the k+1th and all previous states with more weight given to the most recent ones. A state space model of the system was built from the transfer function and the control was calculated from this model. Thus, the regulator adjusted its output based on a continually updated model of the system. Astrom also included a design method for pole placement of the system with the regulator. Thus, the desired response characteristics could be formed from a regulator. Unfortunately, this method suffered from an enormous calculation load for defining the model and controlling the system.

A learning method for trajectory control of robot manipulators was presented by Arimoto, *et al.*(1984). The process was an iterative method where the kth input was based on the k-1th input modified by an error increment. The scheme essentially consisted of three steps. First a control input was applied, then the error from the desired output and the actual output was found. This error was then used to calculate the change in control input and the new input would be applied to the system. The paper gave a theoretical analysis of the convergence properties of the norm of the error and showed its convergence towards zero as k approaches infinity. A numerical example using a linear system with some unmodelled nonlinear dynamics was used to

show the effectiveness of the algorithm. The error converged to zero rapidly after 5 or 6 steps.

Zhang and Edmunds presented another type of state learning controller. (Zhang and Edmunds, 1992). Their controller was formed as an answer to other model-based adaptive schemes. The controller is constructed without an explicit structure and develops control laws for the system as a mapping of control space to state space. As time progresses, the mapping gains information about the system through state measurements and becomes a more effective controller. The authors used an iterative method where the control input at the kth input is modified by a delta u formed from a sliding mode style equation. Thus, the control first descends from any point in the state space to this sliding surface and once on the surface towards the equilibrium point. The sliding surface was chosen for its mathematical niceties because a convergence condition of the learning algorithm is offered along with the paper. A simulation of the inverted pendulum-cart problem was used to demonstrate the algorithm and its stability. Their tests consisted of bringing the pendulum from a straight downwards position to a full upright one. The tests were successful with the cart oscillating back and forth until it could catch the pendulum and stabilize it. The authors see the implementation of such a method through the use of adaptive neural networks or self-organizing fuzzy controllers.

In concluding this literature review, the historical beginnings of the QIL algorithm are given. The rise of the QIL algorithm comes from an idea published by Michie and Chambers in the late 1960s (Michie and Chambers, 1968). Their idea was to find a machine learning controller which would achieve control using solely the outputs of the

system. The controller would need to evolve through time and also be self organizing. Their solution was a concept known as 'Boxes'. Essentially, this algorithm asks two questions of itself every time step and then proceeds to apply a control to the system. The questions are: Have I been at this state before? and What is currently the best control to apply for this state vector? Immediately, the authors realized that for a computer the answer to the first question is generally "No, I have not been at this exact state." They, therefore, borrowed from Widrow and quantized the state space into discrete regions, hence the term 'Boxes'. The authors then proceeded to test the algorithm with 162 boxes split among 4 states for an inverted pendulum-cart simulation. Their results show the pendulum being balanced in excess of an hour simulated time. Some of the difficulties they had with the algorithm involved what they called 'uninformed' and 'indecisive' boxes. The uninformed boxes were regions where the system rarely went and consequently a control had not been effectively determined. This problem was minor, though, compared to indecisive boxes. These boxes had been visited a statistically significant number of times, but had not developed a control strategy. The authors blamed this on a poor split in the regions which caused a 'don't care' syndrome to develop. Basically either control could be used and the system recovered in a neighboring box. Their suggestion was to give the algorithm a splitting and lumping capability, where it would analyze boxes and if two regions compared favorably a lumping of the regions would occur. In the indecisive boxes, splitting would occur to eliminate the indecision. However, no results on this attack have been published.

In 1983, 'Boxes' was examined again in comparison to neuronlike adaptive elements. (Barto, *et. al.*, 1983). These authors used 'Boxes' as a comparison to an adaptive

neural network they were examining. Their work once again showed the effectiveness of 'Boxes' in learning control of an inverted pendulum-cart system, however, the neural network proved to be a faster learning algorithm and capable of longer balancing times in the same number of learning trials. Unfortunately, both of these papers used system failure as the criterion for updating the 'Boxes' algorithm. This method does limit the amount of learning to be gained from any one learning trial.

In 1993, the 'Boxes' algorithm was extended and used for control of an experimental testbed (Kiel, 1993). Kiel's thesis examined the viability of inductive learning methods for vibration control with the possibility of Active Damage Control being achieved from this research. His testbed was a simply supported aluminum beam with a single harmonic disturbance. The algorithm then used a state of three position measurements through time to locate a box and learn a control. The algorithm was modified to use a performance index of the average of the absolute value of strain over the last three time steps to evaluate the effectiveness of a particular control. The experiment achieved moderate success. The biggest problem appeared to be a lack of sufficient information to determine the phase of the disturbance in order to achieve disturbance compensation.

Kiel's work is the springboard for the creation and investigation of the QIL algorithm. His work showed the potential for an AI method in active damage control. However, several issues were left unanswered or as recommendations for further work. In the next chapter, these issues and others to be analyzed in this work, are raised and examined.

CHAPTER 3

THEORY AND ISSUES

The investigation of the QIL algorithm is motivated by the desire for Active Damage Control (ADC). The research into ADC is relatively new. This chapter will examine the general issues associated with ADC. After the big picture overview, the subsequent research issues involved with the QIL algorithm and their relation to ADC will be explained.

3.1 Research Issues for ADCS

As previously mentioned, the research in ADC revolves around two central areas, damage control and damage prevention. These two areas present numerous problems for investigation. As a first step, the term damage needs to be defined as it relates to ADC.

The Random House dictionary states damage is "injury or harm that impairs value or usefulness". Paraphrasing directly for ADC, damage would be a change in system dynamics that impairs its desired performance. However, this area of study also includes the intent that such a change will occur due to some event acting on the system. Thus, a refinement of the definition of damage for ADC would be a change in system dynamics due to an event that impairs desired performance. The source of the event does not need to be known, although obviously knowing the source of change is

helpful. The class of problems covered by this interpretation remains vast. The obvious cases of impact damage to a structure or fatigue failure within a mechanism immediately spring to mind, but by extending the domain of thought, any time varying system also fits the definition. It should be noted that system performance is not always impaired by a change in system dynamics. In these cases, 'damage' has not occurred to the system. In general, ADC will control a system and then if damage occurs, proceed with appropriate measures to adapt the control law to the new system. Ideally, the control strategy would prevent damage from occurring and inhibit the growth of any damage in the system.

The prevention of damage is a separate area from damage growth control. The goals for damage prevention work around the precept that harmful effects can be controlled. It is possible to achieve damage prevention, but impossible to prevent any damage from ever occurring to a system. Gremlins always seem to find a way to work their magic on parts, even for the best designed system. Consequently, this area of research needs to address what type of damage can be prevented? Fatigue failure is a common occurrence in many types of rotating machinery and recently active methodologies have shown the ability to increase fatigue life. Are there other types of damage that can be similarly prevented? The answer is yes; the solution and means still need to be developed. The biggest difficulty in damage prevention is determining where damage will occur and why it will occur? By answering these two issues, an effective method can be implemented which will prevent the damage from entering the system. The damage prevention area requires an understanding of the routes damage can enter the system. By proper modeling and knowledge of potential damage sites in the design of

a system, the location of sensor, actuators, and other hardware can be properly placed to counter the effects which cause damage.

The other issue is active damage growth control. The concept is to contain damage in a local region and prevent its spread in the system. The main issues are how does damage grow and what needs to be done to prevent its growth? A model for the mechanism of damage growth helps the designer understand the physical interactions damage presents to the system, but it is not necessary for it to be available. Many methods can use an internal representation with no physical meaning to great effect. However the method is modeled, it is important to include its effect on the system. Only with knowledge of the damage's interaction can a means to contain it be effected.

The methods for preventing damage growth are not immediately obvious. The biggest difficulty being the requirement for *a priori* knowledge of how the damage occurs. Consequently, many of the control methods will be adaptive in nature and be robust enough to handle a variety of situations. Issues involved for accomplishing this goal revolve around actuator dynamics, location, available force, power requirements, etc. The problem again boils down to answering where damage enters the system and controlling that route in such a way to contain the damage to a localized area.

3.2 Research Issues for the QIL algorithm

The QIL algorithm is being researched as an Active Damage Control strategy for controlling the growth of damage. The research is driven by the goal of a model-independent controller. The definition of model-independent in this case is no

knowledge of the plant or its dynamics is used in the design process. The advantage sought for ADC is a robust controller which will adapt its internal control laws as the system changes.

The issues involved with studying the algorithm revolve around its learning process. The first question is: "Does QIL behave like any other control method, specifically state feedback?" The reason being if an analogous method to QIL is found, then analysis of QIL should be easier. The specification of state feedback is from the fact the algorithm is a feedback method of control and it uses the outputs or states of the system.

In analyzing the QIL algorithm, many more questions were asked to assess its effectiveness as a control method. Previous studies showed the performance index (PI) is the key to the learning process (Pascoe, Robertshaw, and Kiel, 1994). Consequently, the selection of an effective PI became a primary concern. Associated with the PI, is the issue of this adaptive search technique avoiding the local extrema in the state space. Ideally, a proper choice of the PI should avoid these problems. In addition, the type of control problems QIL can master are not known. If there are limitations to QIL effectiveness, it is beneficial to find them.

Since QIL quantizes the state space and the control domain, the nonlinearities associated with the quantization needs analysis. Is it possible too many or too few quantizations impair the algorithm's learning ability? The selection of how the quantizations are divided may create an uncontrollable problem. In addition, the system is a digital implementation, the variation of the sample period may affect the

amount of quality information passed to the algorithm. Is there an optimal sample rate? What is the relation between sample rate and quantizations?

The issues raised in this section are the core thrust of the research into the QIL algorithm. They are not a complete list for a total analysis of QIL, but provide an appropriate start to evaluate its effectiveness for controlling systems especially those expected for ADC.

CHAPTER 4

THEORETICAL ANALYSIS OF QIL

This chapter presents the outline of the QIL algorithm and the extensions made from previous versions for improved performance. An analysis of the learning process of QIL is used to answer some of the issues raised in the last chapter as well as provide an insight into understanding the algorithm. Finally, the disturbance compensation problem is examined and the necessary information QIL will need to effect control of the problem is given.

4.1 The QIL algorithm

The QIL algorithm is organized as an intelligent search program that learns from the examples (i.e. system inputs and outputs) presented it. Initially, the algorithm possesses little or no knowledge of the system to be controlled. As time progresses, however, heuristics or rules are developed and used to control the system. This section will show how these rules develop and the factors that can influence their development.

The search pattern is built around the following four core steps:

1. Identify the quantized region the system states or outputs occupy.
2. In this region, find the control with the best performance value from the control domain.
3. Apply the control and calculate the associated performance index.
4. Update the control domain for the region with the new performance index.

These four steps are iterated for each example and the control rules develop from the constant updating of each control in the control domain. Accordingly, the performance index (PI) determines the rules which develop in each region. Generally, the PI is a mathematical expression which forms a penalty function of the states and the control. In the early stages of learning, the algorithm moves like a random walk in the state space because there is no knowledge of the system. As more examples occur, the algorithm weeds out the poor performers from the better ones. Eventually, after enough examples, a control from the control domain is selected as the best for a specified region of the state space. Another perspective on the process is QIL forms a mapping from the quantized state space onto the discrete control domain. The control then is a function of the state space ($u = f(x)$), albeit a piecewise continuous one. In effect, QIL becomes a table, which is referenced in each region of the state space, for control values.

In the framework of Artificial Intelligence, the search pattern described above is classified as a learn-by-examples approach or an inductive learning method. The algorithm generates a control function based solely on the outputs generated from specified inputs using a guiding heuristic to achieve a desired goal. Since the algorithm does not rely on a model for finding the control function, the method is also a model-independent controller. An important observation of the whole approach is that the controller is only as good as the information given it. If the examples are poor representations of the system or do not provide adequate information about the system dynamics, then the resulting control will generally be poor as well. This characteristic is caused by the inductive learning methods used to generate the control function.

Inductive learning methods assume the examples presented represent a broad spectrum of the possibilities and the conclusions drawn from the examples will be correct.

The preceding paragraphs present the outline of the QIL algorithm and its learning process. In analyzing the details of QIL, it is readily apparent QIL is a nonlinear control method. The main cause of the nonlinearities are the quantizations of the state space and the control domain. The difficulties of nonlinear systems prevent a simple expression of QIL's learning process being made. Using the graphical techniques found in nonlinear analysis allows some insight into the inner workings of QIL.

In understanding the QIL algorithm, it will be best to start with a general linear, time-invariant, second order system of the form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{1}$$

where $\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$ $\mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$ $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

First consider the homogeneous solution to equation 1 (i.e. $\mathbf{u}(t)=0$ for all t). It is of the form:

$$\mathbf{x}(t) = \mathbf{C}e^{\mathbf{A}t} \tag{2}$$

with **C** a constant vector and the exponential term equivalent to the matrix exponential function. The solution has five distinct structures which depend on the eigenvalues of **A**. The five cases are:

1. Real unequal eigenvalues of the same sign
2. Real eigenvalues of opposite sign
3. Equal eigenvalues
4. Complex eigenvalues

5. Pure imaginary eigenvalues

Using the phase plane (plotting x_2 vs. x_1), a graphical insight into the system dynamics for each case will be found (Boyce and DiPrima, 1986). Assuming the $\det \mathbf{A} \neq 0$, then the critical point of the system is the origin ($\mathbf{x} = \mathbf{0}$) and a stable system will move from any initial condition to this point. A stable system is defined by the real component of the eigenvalue. If the real part is nonpositive, then the system will be stable. Thus, case 2 will never be globally stable. For this discussion, all the systems will be assumed stable where possible.

The general homogeneous solution to equation 1 is:

$$\mathbf{x}(t) = c_1 \mathbf{v}_1 e^{\lambda_1 t} + c_2 \mathbf{v}_2 e^{\lambda_2 t} \tag{3}$$

where $\mathbf{v}_1, \mathbf{v}_2$ are eigenvectors corresponding to the eigenvalues of λ_1, λ_2 respectively. The coefficients are determined from the initial condition vector. Case 1 corresponds physically to an overdamped system. In the phase plane, the result is termed an improper node. From an arbitrary initial condition, the trajectory will asymptotically approach the origin along the line of the eigenvector with the larger eigenvalue. The exception is if the initial condition lies on either line described by an eigenvector, then the trajectory will move along the line towards the origin. Case 2 forms a saddle point in the phase plane. The positive eigenvalue generates a term that tends to infinity in equation 3 as time approaches infinity while the other term tends toward zero. Thus, an arbitrary initial condition will asymptotically approach the line described by the eigenvector of the positive eigenvalue. The only stable solution will occur if the initial condition lies on the line described by the other eigenvector. In this case, the trajectory follows the line into the origin because the coefficient of the unstable term is zero. The third case contains two possibilities. If the repeated eigenvalue has two independent

eigenvectors, then a proper node is formed. For a proper node, all the trajectories follow straight lines in to the origin from any initial condition. If the eigenvectors are not independent, an improper node results which has the same characteristics of case 1. Case 4 corresponds physically to an underdamped system. The trajectories formed are spirals in to the origin. Using polar coordinates, the trajectory is described by the equations:

$$r(t) = ce^{at}$$
$$\theta(t) = -bt + \theta_0$$

where a and b are defined by the complex number $a \pm ib$ corresponding to the eigenvalues of the system. The constants are determined by the initial conditions. The spiral can rotate clockwise or counterclockwise and be skewed depending on the elements of matrix **A**. Case 5 is then a special case of case 4 with a = 0. Then, the trajectories are ellipses centered on the origin and described by the equation:

$$a_{21}x_1^2 + 2a_{22}x_1x_2 - a_{12}x_2^2 = k$$

The initial conditions determine the constant k which defines the radius of the ellipse.

Having determined all possible trajectories in the phase plane for the homogeneous solution, consider the effect of **Bu** if u is some arbitrary constant. Equation 1, then is a system of first order differential equation with the forcing function **Bu**. The critical points of the nonhomogeneous system are found by setting $\dot{\mathbf{x}} = \mathbf{0}$. Equation 1 becomes:

$$\mathbf{0} = \mathbf{Ax} + \mathbf{Bu} \tag{4}$$

Solving equation 4 for **x** gives,

$$\mathbf{x} = -\mathbf{A}^{-1}\mathbf{Bu} \tag{5}$$

The inverse of **A** is known to exist because it was assumed $\det \mathbf{A} \neq 0$. Therefore, equation 5 shows the critical point of the system to be a constant vector which is not

the origin. The five cases discussed above will still hold since they are only dependent on the eigenvalues and eigenvectors of **A**. However, instead of the trajectories moving towards the origin, they will now move towards the critical point described in equation 5. The only constraint is that the system of equation 1 be controllable.

The above discussion is relevant to QIL since QIL chooses from a discrete control domain. Once QIL finds the best control value for a region, then every time when that region is hit, QIL applies this control. Consequently, the system follows its open loop dynamics with the critical point defined by equation 5 for the specified region. At the boundary of the quantized region, a new control value will be applied and the system will move towards the new critical point defined by the control with an initial condition defined by the point where the boundary is crossed.

Using the example of an undamped system, the learning process can be explained more completely. First note an undamped system has pure imaginary eigenvalues and thus falls under case 5. If the matrix **A** is selected with the form $\begin{bmatrix} 0 & \mu \\ -\mu & 0 \end{bmatrix}$ then the phase plane trajectory is a circle with radius equal to distance from the circle center to the initial condition vector. Using equation 5, the center of the circle is defined as:

$$\mathbf{x} = -(\mathbf{A})^{-1}\mathbf{B}u$$

$$= -\begin{bmatrix} 0 & -\frac{1}{\mu} \\ \frac{1}{\mu} & 0 \end{bmatrix}\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}u$$

$$= \begin{bmatrix} \frac{b_2}{\mu} \\ \frac{-b_1}{\mu} \end{bmatrix}u$$

If u = 0, then the origin is the center of the circle as expected. As an alternative method to verify this result, solve equation 1 with u as a constant. The system to solve is:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & \mu \\ -\mu & 0 \end{bmatrix}\mathbf{x} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}u$$

or (6)

$$\dot{x}_1 = \mu x_2 + b_1 u$$

$$\dot{x}_2 = -\mu x_1 + b_2 u$$

Using the fact $dx_2/dx_1 = (dx_2/dt)/(dx_1/dt)$, equation 6 becomes:

$$\frac{dx_2}{dx_1} = \frac{-\mu x_1 + b_2 u}{\mu x_2 + b_1 u} \tag{7}$$

Rearranging 7 gives:

$$(\mu x_2 + b_1 u)\, dx_2/dx_1 + (\mu x_1 - b_2 u) = 0 \tag{8}$$

Equation 8 is a first order differential equation which is exact. Using the theory for solving exact equations, a function $\Psi(\mathrm{x}_1, \mathrm{x}_2)$ is found to be:

$$\Psi(\mathrm{x}_1, \mathrm{x}_2) = (\tfrac{1}{2}\mu x_1^2 - b_2 u x_1) + (\tfrac{1}{2}\mu x_2^2 + b_1 u x_2)$$

and the solution for equation 8 is:

$$C_1 = (\tfrac{1}{2}\mu x_1^2 - b_2 u x_1) + (\tfrac{1}{2}\mu x_2^2 + b_1 u x_2)$$

By completing the square of both parenthetical quantities and changing the constant C, the solution is an equation for a circle centered at:

$$x_1 = \frac{b_2}{\mu}u \qquad x_2 = -\frac{b_1}{\mu}u$$

This is the same result predicted by equation 5. Again the radius of the circle is determined by the initial conditions of the problem.

Therefore QIL will control an undamped system by tracing circles centered about each discrete control value. Since the state space is quantized into different regions and only one control becomes the best for a region, a phase plane graph of a QIL controller will show arcs of circles in each region. The center of each arc will be described as above and based on the value u chosen.

The above example can be extended to any of the five cases for second order systems. The critical point will serve as the focus for the trajectories and as different controls are chosen, the point will move as described by equation 5. One viewpoint is QIL jumps from one trajectory to another as it crosses the boundaries of the regions, relying on the open loop dynamics to achieve its objective for a specific region. The learning process is then to find the trajectory which minimizes the penalty function for all states within a specific region.

So far, the analysis has covered a single-input/multiple-output, second order system. The arguments made about QIL's learning process, however, do not rely on the order of the system. The solution to the homogenous part of equation 1 will always be equation 2, independent of the system order. Now however, the possibilities of multiple pairs of complex eigenvalues, real and complex eigenvalues, and eigenvalues

of multiplicity k will occur. For an n^{th} order system, the phase plane becomes an n-dimensional hyperspace in which the system trajectory travels. The trajectory will be described by a hypersurface determined by the eigenvectors of **A**. For a single input system, the critical point will still be defined by equation 5. Thus, QIL will still jump from trajectory to trajectory, but now the hypersurface is nearly impossible to see graphically.

The multiple-input/multiple-output case increases the complexity of the problem as well. The critical point is now determined by the vector **u**. Consequently, QIL will need to search both control domains to find the best combination of controls for a specified region. A task still possible to accomplish, but now the computations necessary for the learning process to converge expands exponentially. This is very similar to the "curse of dimensionality" suffered by dynamic programming.

In implementing the QIL algorithm, the system of equation 1 will be a discrete time system. The main reason for this change is the mathematical expression for the performance index is calculated using sampled points from the system. Thus, the system is sampled with a zero-order hold and equation 1 becomes a difference equation of the form:

$$\mathbf{x}_{k+1} = \Phi\mathbf{x}_k + \Gamma\mathbf{u}_k \tag{9}$$

Using equation 9, an argument can be made that QIL will converge to a state feedback as the spacing of the quantizations of the state space become infinitesimal and the number of controls becomes infinite.

The proof of the above paragraph is as follows. First constant state feedback is formed by $\mathbf{u}_{k+1} = f(\mathbf{x}_k) = -\mathbf{K}\mathbf{x}_k$ where **K** is a gain matrix chosen to create a desired set of eigenvalues of the closed loop system. For the discrete time implementation of equation 9, the control is held for one sampling period and then the next control is calculated. While the control is held, the continuous system runs with its open loop dynamics with the constant forcing term of **Bu**. If QIL quantizes the state space into regions as small as the quantization error of the computing system, then every region would describe only one point. Quantizing the control values to the same threshold, would provide an infinite choice of discrete values. With the proper performance index, QIL would then find the one control value corresponding to the one state value that satisfies $\mathbf{u}_{k+1} = -\mathbf{K}\mathbf{x}_k$. Once the system moves to the next region a different control value would be chosen and thus the mapping of the state space to the control space would occur. A physical realization of this system would be impossible however, because the number of computations to evaluate the infinite number of controls for the infinite number of quantization regions is infinite. Accordingly, an infinite amount of time would be required to search all the possibilities. This argument, though, shows QIL behaves like state feedback except the grid formed by the quantizations of the state space is extremely coarse and the performance index behaves like the gain matrix **K** instead.

As alluded to previously, QIL suffers from its own version of the "curse of dimensionality". The number of calculations QIL performs is directly proportional to the time spent in converging to a solution. The number of calculations QIL will perform can be estimated by multiplying the number of controls in the control domain, m, by the number of quantized regions in the state space, q. This number, call it x, will

be the absolute minimum number of calculations QIL can perform and achieve an answer. However, several factors still need to be considered. The performance index is a formula which generally uses a set number of points, p, in its evaluation. Then, x must be multiplied by the number of points for a revised minimum. In addition, the system will generally not move through a region in only m steps. These extra steps are an unknown variable, but by using the approximation of the distance of the shortest boundary line divided by the sample period, the minimum number of steps a system can move through a region is derived. Using this value and multiplying it with the revised minimum described above, a new minimum is found. As an example of the magnitude of this minimum, assume a second order system is quantized into a 16x16 grid with 9 controls. The performance index will use 10 examples for its function evaluation and the minimum number of steps through a region is approximately 20. The resulting number of calculations QIL will have to perform will be $16 \times 16 \times 9 \times 10 \times 20 = 460{,}800$. This assumes QIL only uses the bare minimum number of calculations to converge. In practice, the minimum has been found to be 2 or 3 times the bare minimum. Consequently, QIL is a computationally intensive algorithm. It becomes advantageous then to try to reduce the size of the grid and the number of controls while still maintaining the ability to control the system. Since the goal for QIL is to be a model-independent controller, optimizing the grid size versus maintaining the control of the system proves to be a difficult problem.

A method to establish a grid knowing the system model is possible and using some of the steps from this development can provide general guidelines for a grid when the model is unknown. Since QIL is similar to state feedback, a good starting point for developing the grid is with the optimal state feedback answer. The optimal answer is

derived from the linear quadratic regulator using the cost function, $J(\mathbf{x},\mathbf{u}) = \int_0^\infty \mathbf{x}(t)^T \mathbf{R}\mathbf{x}(t) + \mathbf{u}(t)^T \mathbf{Q}\mathbf{u}(t)\, dt$. The answer is state feedback with a gain matrix, **K**. The optimal control is then **u = -Kx**. QIL, though, has a finite number of controls available. Substituting each of these values into the control function, the equation then describes a hypersurface in the hyperspace of the vector **x**. For a second order system, the solutions are lines described by, $x_2 = -(k_1/k_2 * x_1 + u/k_2)$. If the quantization boundaries are made where the line intersects the axis, a grid which will be controlled by QIL is set up. Figure 1 shows the graphical interpretation of this idea.

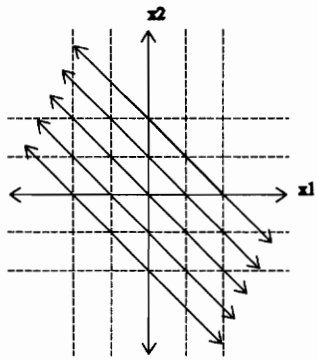

Figure 1:Phase Plane Plot

The dotted lines represent the boundaries of the boxes in the state space. If a box only contains one line, then it follows that a performance index that acts like the cost function for the linear quadratic regulator will select the control corresponding to the line for the specified box. Thus, by using the knowledge of the model, a grid can be formed which has one control in each region that is the optimal one of the set.

If the model is unknown, then there will not be a predictable solution guaranteeing the one optimal control per region criteria outlined above. In this case, the designer needs

to rely on intuition and the art of engineering. Some guidelines to use can be derived from knowledge of the outputs or sensed measurements of the system. The number of outputs determines the minimum order of the system. Accordingly, the hyperspace corresponding to this minimum system order is known. In addition, it has been shown there are hypersurfaces which correspond to constant control values in the hyperspace. If an assumed shape of the hypersurface, like **u=-Ky**, is used, then the intersection of this surface with each axis can be calculated for a constant value of u. Thus, an initial grid can be formed and used in the QIL algorithm. The usefulness of the grid, however, needs to be tested by running the QIL algorithm and evaluating the solution. Factors which will affect the answer are the performance index, the sampling period, time delays, and the accuracy of the assumed hypersurface in relation to the system. The method is then iterated by tweaking the number and spacing of the quantizations as well as the factors mentioned above until a satisfactory response is found.

Another factor which is a common problem in digital systems is time delays. The time delays generally cause problems with controllability and achieving desired response characteristics. For QIL, this problem does not really exist. The algorithm does not access a model of the system and thus, time delays end up being incorporated as a characteristic of the system. The control function is generated with time delays factored into the performance index implicitly. The sample period functions in the same manner as time delays. It is a characteristic of the system and consequently QIL learns its control function without considering the sample period. However, if time delays or the sample period cause the system to be uncontrollable or seriously degrade the response of the system, then QIL may be unable to find a control function which achieves the desired performance. The problem can be slightly compensated in the QIL

algorithm by changing the number of controls, the quantization levels, or the sample period, but these measures are not generally sufficient in dealing with the real problem which is the digital system's characteristics. Thus, unstable systems or conditionally stable systems are not ideal candidates for control with QIL because the algorithm can drive the system unstable or out of its control authority range before it learns enough about the system.

4.2 Extensions Made to QIL

The extensions to the QIL algorithm are made to improve its performance over previous versions. The key factor in previous studies was the performance index. It was found that the algorithm's response to different problems depended greatly on the nature of this mathematical expression. The first change made to the PI is changing the expression to an LQR-style cost function because of QIL's similarity to state feedback. In this study, the PI is chosen to be $PI = \sum_{i=0}^{N} \mathbf{x}(i)^T \mathbf{Q}\mathbf{x}(i) + \mathbf{u}(i)^T \mathbf{R}\mathbf{u}(i)$. The matrices **Q** and **R** weight the state values and N is the number of time steps the algorithm uses to determine a control value's fitness. The QIL algorithm then examines each control value and uses the value which produces a minimum. The only difficulty with the PI as expressed above, is the tendency for the algorithm to be trapped by local minima in the state space. In addition, as the algorithm is learning, there is a possibility that the best choice is eliminated prematurely by a poor control sequence resulting from the trial and error search. The PI is modified with an averaging process that reduces the effect of the initial randomness of the search. Another extension is an idea borrowed from the theory of Genetic Algorithms. The concept of mutation in Genetic Algorithm

(Goldberg, 1983) is used to admit possibilities which may have been omitted in the initial population or eliminated in cross-breeding. The goal being to avoid entrapment by local extrema in the function space. The use of mutation in the QIL algorithm follows in the same vein. The mutation factor is used to average a zero into the PI of a randomly chosen control value. The chance for a mutation is small (generally <1%) and the effect is small as well. The main idea is to allow the possibility for some controls to be tried again to reexamine their effectiveness. Also, the LQR-style performance index uses all observable state information in determining the penalty. The advantage being a better control is effected and the process will converge slightly faster. In addition, the idea of localized control is a possibility with the LQR-style performance index because the weighting matrices can be selected to penalize certain states or outputs more than others.

The control domain is also changed to values which have enough authority to control the system as well as have enough discrete points to provide control. Previous versions suffered from a serious lack of control choices and consequently could not effect an adequate control. In addition, the control values are randomly chosen in the initial stages. The probability of a very poor control sequence affecting any one control is avoided with this method.

4.3 Disturbance Compensation with QIL

Disturbances are a common problem in controls engineering. Feedback methods of control generally require an accurate model of the disturbance to cancel the disturbances effect from the dynamic response of the system. Such methods are not

very robust and work only for a narrow-band disturbance rejection. The more common approach is to include some type of feedforward path in the control design which cancels the effect of the disturbance input. The advantage of this method is that combining it with state feedback allows disturbance compensation along with pole placement. The block diagram of both methods is outlined below.

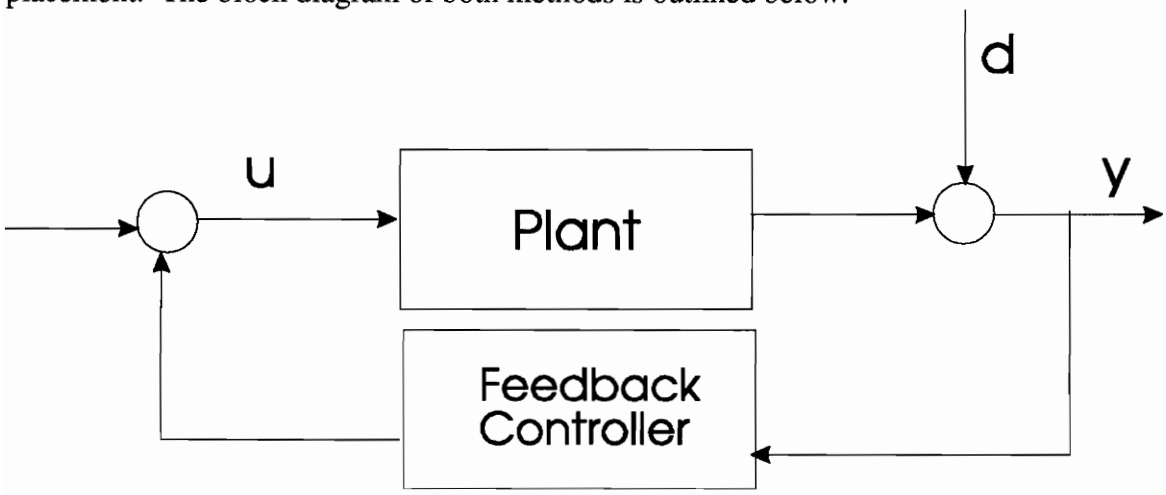

Figure 2: Block Diagram

Theoretically, QIL should compensate for the disturbance. The requirements for the algorithm to achieve this objective is a measurement of a variable related to the disturbance. The measured variable will need to be quantized into regions and these areas add an extra dimension to QIL. The variable can be something like a magnitude of the disturbance signal or the phase of the disturbance. Since QIL is computationally intensive, in general only one measurement should be used and usually it is enough. The measurement is necessary because QIL needs to incorporate this extra variable into its internal system transfer function. Without knowing something about the

disturbance, QIL cannot converge to an answer. The measurement for the disturbance can also come from an estimator if it is not possible to physically sense the disturbance.

CHAPTER 5

RESULTS AND DISCUSSION

The results from computer experiments performed with the QIL algorithm are presented in the following three sections. First, the initial condition problem is used to demonstrate the concepts of Chapter 4. The second section ascertains QIL's effectiveness for disturbance compensation. Finally, the results of QIL's control with time-varying systems are shown.

5.1 Initial Condition Problems

The initial condition problem is a good starting point for studying the learning process of QIL. The simplicity of the problem allows the design parameters of QIL to be effectively isolated for analysis. In addition, solving this basic problem allows the strengths and weaknesses of the algorithm to be found.

Another advantage in studying the initial condition problem is the ability to extend the results to the general tracking problem. By viewing the initial condition problem as a special case of the tracking problem with the reference being zero, then the results can extend to tracking any reference.

The final advantage in studying the initial condition problem is the common occurrence of this problem in the history of controls. A variety of methods have been used to find

a solution. The wealth of knowledge can be tapped to compare QIL to traditional solutions. The comparison allows a measure of QIL's effectiveness to be obtained and used in determining the feasibility of QIL in various applications.

The first group of experiments used the classic undamped single mass-spring system sliding on a frictionless surface (Figure 3). This linear system is convenient for study because it has simple difference equations which are ideal for numerical simulation. Also, the phase plane can be used as a graphical tool for analysis.

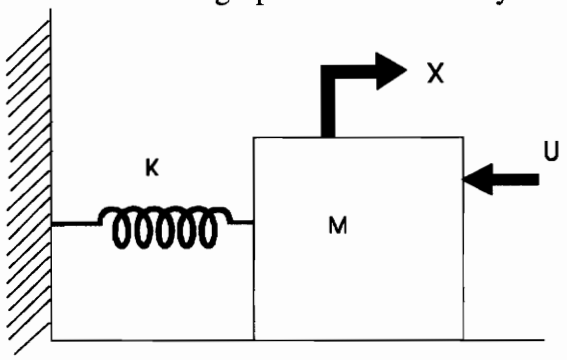

Figure 3: One Mass System

The experiments were designed to examine three factors: the Performance Index, the size of the quantization grid, and the sample period. First, an assortment of PIs were examined. The basis for each PI evolved from the kernel function of the Linear Quadratic Regulator (LQR) cost functional (i.e. $\int \left(x^T Q x + u^T R u \right)$). The second set of experiments studied the effects of varying the size of the quantization grid from fine to coarse. The final set of experiments analyzed the effect of changing the sample period of the system.

The choice of these three factors follows from the theory discussed in Chapter 4. As previously discussed, the PI appears to be the key to the learning process of the algorithm. Identifying the characteristics of a PI which succeeds in control is of primary importance. The quanization grid and sample period affect the learning process as well. QIL requires examples to converge upon an answer and the number of examples required is based upon the size of the grid. The sample period affects the algorithm by presenting varying lengths of operation of the open-loop system, thus the algorithm's control laws should vary with the length. Demonstrating the difference in control provides more insight into the mechanism of the algorithm's learning potential.

The initial studies into different PIs necessitated the formation of a metric to evaluate the merit of each PI. The effectiveness ratio was formed for this purpose. It is a ratio of the cost generated by a PI to achieve control over the cost for the optimal solution of the problem. The effectiveness ratio, ε, is:

$$\varepsilon = \frac{Cost_{QIL}}{Cost_{Optimal}} \tag{10}$$

For these experiments, the optimal solution was determined using the state feedback gains from LQR. The choice of LQR as the optimal solution seems logical. QIL uses the states of a system to generate a control value. The process is nearly identical to state feedback with the difference being the gains for state feedback are calculated beforehand. Thus, it seems natural to use the optimal state feedback answer as a yardstick to measure QIL.

The experiments for examining performance indices analyzed 3 different formats. The first format was a summation using only the LQR kernel. The second format was an

recursive averaging function. In this formula, the LQR kernel was used to calculate the penalty cost which was then averaged with the preceding cost for the box. The final format used a weighted average method. The LQR kernel calculated the penalty cost. The value for the box was then determined by taking the new cost and adding it to four times the old cost divided by five. The result was a function which adjusted the cost of the box using twenty percent of the new cost.

The formats were then modified by summation length for ten or twenty points of data. The longer summations provided QIL with more data points to generate a penalty. Consequently, if a control signal led to a more favorable position over a longer time, then the box would reflect this characteristic and be more likely to use the same control again. Another variation in the formats used the mutation concept from genetic algorithms. Initially when QIL is learning a system, poor control choices can be made after what is known by the designer to be a more favorable signal. The consequence is a favorable signal may never have another chance to indicate its superior status. Mutation allows these signals to gain a second chance. The disadvantage with this method is more examples must be presented to the algorithm.

The experiments analyzed four variations of three performance indices. The experiment was to control the single mass system with 9 controls and a ten by ten grid of boxes covering position and velocity. The grid was designed using the method outlined in Chapter 4. Two hundred trials for five hundred points were used for a total of 100,000 examples presented to the algorithm. The results are shown in Table 1.

Table 1: Effectiveness Ratios for Performance Indices

Performance Index	ε	Calculated Cost	Control Effort
$\int_0^\infty \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}$	1.00	145.3	45.9
$\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i$	2.07	301.1	135.0
$\left\{\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i\right\} \Big/ N$	1.70	246.3	77.5
$\left\{\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i + 4*\text{P.I.}\right\} \Big/ 5$	1.66	241.7	79.5
$\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i$ w / mutation	1.83	266.3	91.6
$\left\{\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i\right\} \Big/ N$ w / mutation	1.81	262.9	57.8
$\left(\sum_{i=0}^{10} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i + 4*\text{P.I.}\right) \Big/ 5$ w / mutation	1.72	249.8	76.4
$\sum_{i=0}^{20} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i$	2.16	313.5	101.1
$\left\{\sum_{i=0}^{20} x_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i\right\} \Big/ N$	1.83	265.3	76.6
$\left\{\sum_{i=0}^{20} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i + 4*\text{P.I.}\right\} \Big/ 5$	1.84	267.5	103.7
$\sum_{i=0}^{20} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i$ w / mutation	2.08	302.5	115.2
$\left\{\sum_{i=0}^{20} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i\right\} \Big/ N$ w / mutation	1.70	246.5	65.6
$\left(\sum_{i=0}^{20} \mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i + \mathbf{u}_i^T\mathbf{R}\mathbf{u}_i + 4*\text{P.I.}\right) \Big/ 5$ w / mutation	1.60	232.5	63.7

The best performance of QIL, $\left(\sum_{i=0}^{20}\mathbf{x}_i^T\mathbf{Q}\mathbf{x}_i+\mathbf{u}_i^T\mathbf{R}\mathbf{u}_i+4*\text{P.I.}\right)\Big/5$ w / mutation generates an effectiveness ratio of 1.60. This ratio indicates QIL incurs 60% more cost than the optimal system. Using a phase plane plot, the source of most of the extra effort comes from the chattering about the origin.(Figure 4) As previously stated, the control solution QIL derives uses only one control for a quantized region. The chattering is the result of this characteristic. By examining the graph, the boundaries of the grid can be clearly seen as points where the control signal is switched and the radical change in the system's profile. The result is an increase in control effort and state penalty. Thus, a cost increase is expected.

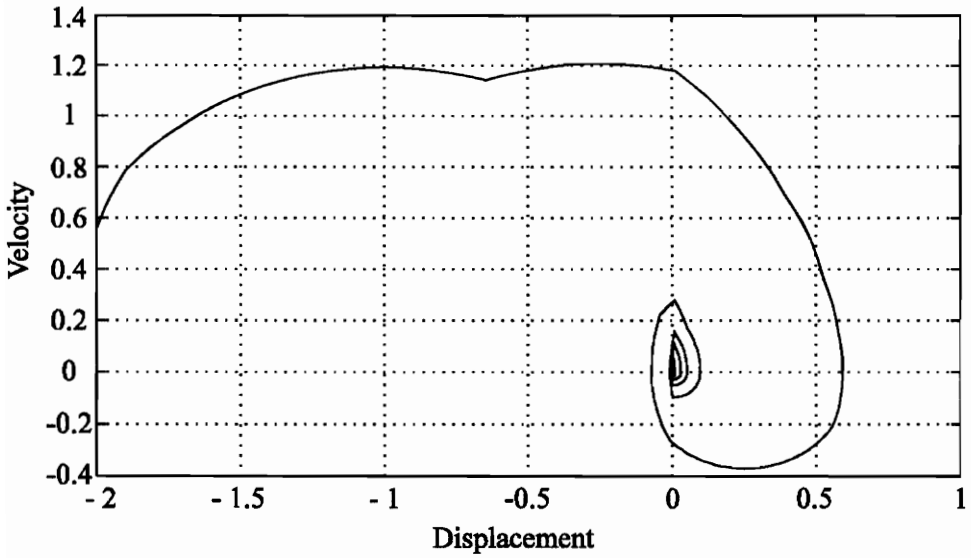

Figure 4: Phase Plane Plot of One Mass System

The next design parameter investigated was the quantization levels. The quantizations are important for several reasons. First, the size of the quantization grid affects QIL's performance and the control effected. Second, the effectiveness of a PI is dependent

on the quantization grid. Table 2 shows the effectiveness ratios for various quantizations of the state space. One fact not apparent from the table is that all the tests ended near the origin indicating that all the systems became stable.

The higher ratios for the coarser grids are caused by one control being applied over a larger region or longer time period. The fine grids result in better performance and a smoother control. Unfortunately, the expanded grid also means more examples are needed to find a solution. Table 2 shows the effect when the grid expands to be 16x16 in size. All the grids were exposed to the same number of examples. Thus, the 16x16 grid is not exposed to enough knowledge to achieve the lower cost expected from it. The table indicates a grid size of 10x10 is the optimum of this set. The 10x10 grid is based on the design philosophy of Chapter 4 where the grid was designed for one control per region.

Table 2: Effectiveness Ratios for Various Quantization Grids.

Quantization Grid	Effectiveness Ratio (ε)	
4 Regions (2x2)	2.47	1.74
16 Regions (4x4)	2.02	1.71
36 Regions (6x6)	2.41	1.91
100 Regions (10x10)	1.72	1.60
256 Regions (16x16)	1.93	1.95

The final battery of tests on the second-order system analyzed the effects of various sample rates. Two different tests were run for each sample rate. The tests used the best PI and grid quantizations from the previous experiments. Fixing the PI and grid,

the sample rate was changed and the effectiveness ratio was calculated. The second case involved calculating the grid quantizations based on the philosophy of chapter 4. The grid was calculated using the optimal control gains K for each sample rate and the intersection of the lines with the axis used for formulating the grid. The effectiveness ratio for these systems were then determined for the various sample periods. The results for the two cases are given in Table 3.

Table 3: Effectiveness Ratios for Various Sample Periods.

Sample Period (sec)	Fixed Quantization	Recalculated Quantization
1/10	2.26	2.21
1/20	1.60	1.60
1/30	1.75	1.44
1/40	1.71	1.43
1/100	1.54	1.40

The results show that when the grid was designed for the sample period, cost was lower. This indicates there is a relationship between QIL's performance and the design of the quantization grid. A more interesting phenomenon highlighted by this table is the lower cost incurred by faster sample rates. Theory predicts that an infinitesimal sample rate would allow QIL to converge on the optimal answer. The results show QIL is able to reduce its cost when sampled faster. However, a point of diminishing returns may be reached because the fastest sampling rate of 1/100 of a second shows only a 3 percent improvement over 1/40 of a second.

5.2 Disturbance Compensation Problems

The ability to suppress unwanted characteristics will be crucial for an ADC. The disturbance compensation problem is the next logical step in demonstrating QIL's potential for active damage control. Disturbance compensation generally involves the rejection of sinusoidal excitations. The difficulties associated with this basic disturbance problem provides a challenging test for the QIL algorithm.

The modification to the QIL algorithm for the disturbance compensation problem involves adding a measurement of the disturbance. The quantization grid is augmented with a set of discretizations for the disturbance state. The idea is to emulate the more traditional control methods by using a disturbance measurement to adapt the control signal and cancel the disturbance. The experiments were conducted on two systems. The first one used the linear two mass system (Figure 5). The second system was a computer model developed from experimental results of a simply supported aluminum beam (Figure 6).

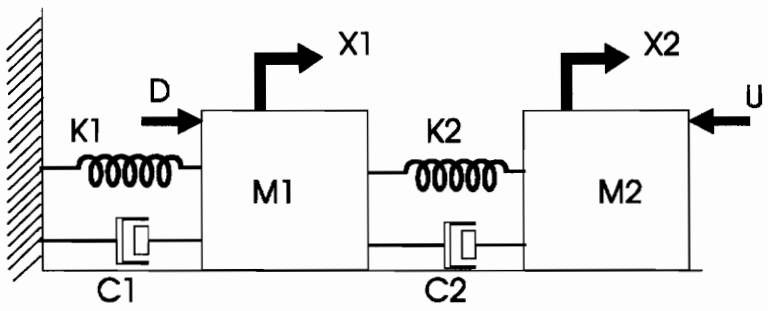

Figure 5: Linear Two Mass System

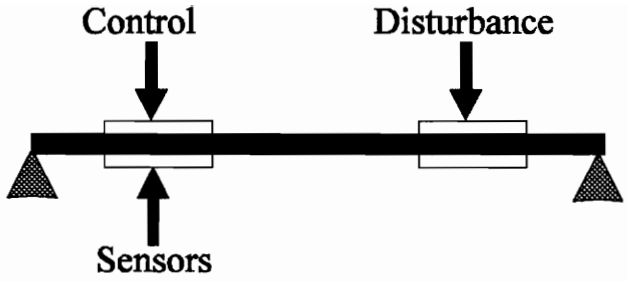

Figure 6: Simulated Simply Supported Beam

The tests for the two mass system used a noncollocated disturbance with respect to the control signal. The disturbance was a sinusoidal signal near the first peak resonance of the system. The experiments investigated two cases, one with a measurement of the disturbance and the other without knowledge of it. The results of these tests show a surprising result. Figure 7 shows the three experimental results. The knowledge case shows a control signal which appears to be a noisy sine wave. This indicates the algorithm used the disturbance signal to generate a control signal 180° out of phase with the disturbance. The same phenomenon is seen for the no knowledge case, but the signal is much more noisy. In both cases, the displacement curve indicates the algorithm did achieve its goal of control of the second mass.

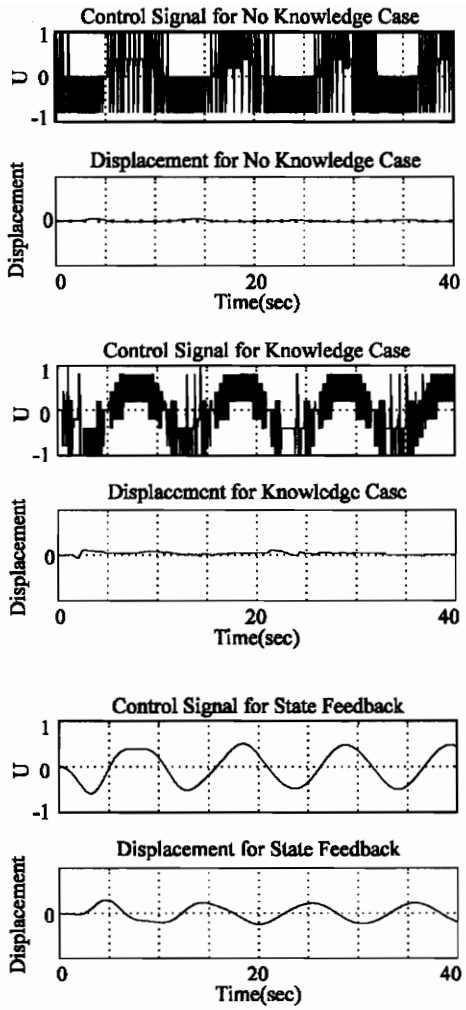

Figure 7: Time Histories of Two Mass Disturbed System

From Table 4, it is seen that with no knowledge of the disturbance or the plant only cost approximately 100 percent more than optimal state feedback answer. With knowledge of the disturbance, the increase in cost is about 50 percent. The result is unexpected. The ability of QIL to discover a compensating control law with neither knowledge of the plant or the disturbance at twice the cost of the optimal solution indicates a surprising ability to learn and adapt.

Table 4: Disturbance Compensation Results

Cost	PI = $\left(\sum_{i=0}^{20} \mathbf{x}_i^T \mathbf{Q} \mathbf{x}_i + \mathbf{u}_i^T \mathbf{R} \mathbf{u}_i + 4 * \text{P.I.}\right) / 5$ w / mutation		State Feedback
	Knowledge	No Knowledge	-Kx
$x^TQx + u^TRu$	371.45	435.04	247.90
uTRu	230.20	291.92	96.75
xTQx	141.24	143.11	151.15

The second set of experiments used a model of a simply supported aluminum beam. The model was obtained from experimental results presented in Fagan(1993). An aluminum beam was excited with a piezoelectric patch near the second peak resonance. The system was identified up to the first five modes and compared with an analytical model. The differences between experiment and theory were explained and compensated in the model. The computer simulation was then used in tests with the QIL algorithm.

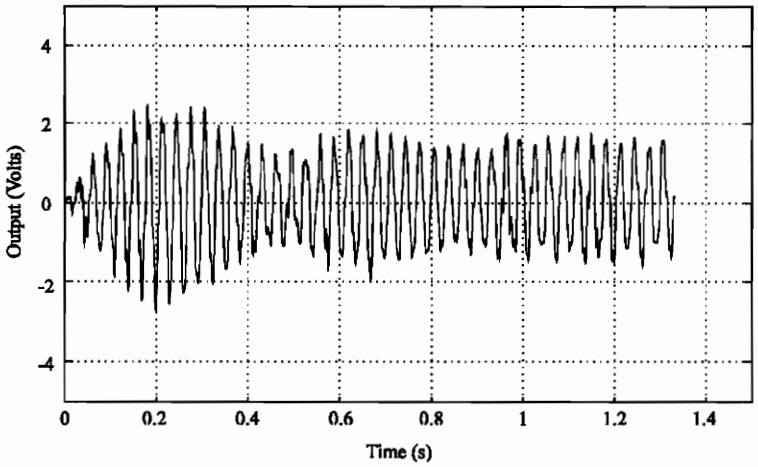

Figure 8: Time History of Simulated Beam

The control developed is disappointing when compared with earlier results. It appears the complexity of the system has resulted in the degradation of performance. The QIL algorithm does not converge on a satisfactory solution. A sample time history plot shows the problems (Figure 8). The algorithm learns a control signal which cannot exactly compensate the disturbance. Consequently the higher modes of the system become excited and QIL provides inadequate compensation. The result is poor vibration control. A similar result was found in the experimental resulted presented by Kiel (1993). His variation of QIL did not significantly reduce the vibration in the simply supported beam. It appears the problems inherent with the discretization of the control domain to fixed points limits QIL's ability to effect an adequate disturbance compensation. Essentially, QIL is forced to pick one control input as the best for a region of the state space. Control overspill can then arise because the fixed input is used for the whole region, unlike state feedback, which has an infinitely variable analog

input. Therefore, higher modes in the beam can be excited by the controller because of an excessive or insufficient control input.

CHAPTER 6

CONCLUSIONS

This thesis' goals were to analyze the learning mechanism of the QIL algorithm and investigate its effectiveness in controlling damaged systems. Both goals were achieved. The following sections will detail the keys to the learning mechanism and the advantages and disadvantages of the QIL algorithm as a controller. Also, recommendations for further work will be reviewed.

6.1 The QIL Algorithm

The QIL algorithm uses inductive learning to generate its control. The control developed is similar to state feedback. QIL selects a control based on the current state of the system and applies this control much the same way the control $\mathbf{u} = -\mathbf{Gx}$ functions for state feedback. The biggest difference with QIL is that the control u is applied over a discrete region or 'box' in the state space whereas state feedback is continuously variable vector.

The key for QIL then is to generate the best control for the region based on the past history of controls used in the region. This process is the basis for inductive learning where past experiences are used to develop the present response to a set of conditions.

In the case of QIL, the past history is generated by the use of the performance index function. The proper performance index will direct the QIL algorithm's learning process by penalizing poor control choices and rewarding good ones until the best choice is found for a region. Consequently, the performance index is the most important design parameter for a QIL controller.

Section 4.2 explains some of the important considerations in selecting a performance index. To summarize that section, the performance index needs to avoid being fooled by local extrema, needs to provide a means to re-evaluate control inputs to compensate for poor control histories, and needs to work on-line with the system.

In addition to the performance index, several other factors need to be considered for the QIL algorithm to function properly as a controller. The factors are:

1. The number of discrete controls.
2. The quantizations of the state space into a grid.
3. The sample period of the system.
4. The sensor inputs of the system.

The number of controls directly determines the effectiveness of the QIL algorithm. First of all the control input possible need to provide the QIL algorithm with enough control authority to control the system. This statement is rather obvious in controls, but since the expectation is that QIL would operate on a system where little or no

knowledge of the system is known, it possible that the selection of controls would not make a controllable system. The second consideration is to provide a broad enough selection of controls. As discussed previously, the greater the number of control inputs, the better control the QIL algorithm can achieve. The disadvantage in increasing the number of controls is a longer learning curve for the QIL algorithm.

The quantizations of the state space into a grid is another necessary design parameter in creating a QIL controller. Two factors in selecting a quantization grid need to be considered. First the number of quantizations to be used and second is the division of the state space by these quantizations. This thesis has shown that one method for selecting a grid uses the number of control inputs to select the grid and the divisions. This method provides a controller which works satisfactorily. For modelless systems, this method does not work well. However, using basic knowledge of the plant and the number of inputs, a first attempt at design can be tried with the stated method. Further work investigating the effect of changing the grid would prove useful in refining the technique of setting a quantization grid.

The sample period affects the QIL controller minimally. As a design parameter, the sample period needs to be chosen such that the system remains controllable. As long as this condition is maintained, QIL can effect control. The shorter the sample period, however, the more effective QIL becomes due to the system becoming a closer

approximation of a continuous system. The QIL algorithm is then able to fine tune itself to achieve the best control for each region.

The final design criteria listed was sensor inputs. The effectiveness of QIL relies solely on the quality of information fed the algorithm. Sensors need to represent the plant and the effect control inputs have on it. During some of the early stages of experiments of this thesis, improper sensor inputs were selected and the results usually ended in unimproved systems or at worse unstable systems. The conclusion is use logical sensor placement for control with the QIL algorithm.

6.2 Disturbed Systems

The experiments on damaged systems consisted of disturbance compensation on two simulated systems. The QIL algorithm proved to be effective on the simple linear two-mass system generating a controlled system with no knowledge of the disturbance or the plant. The QIL algorithm can achieve disturbance compensation for this simple system. The QIL algorithm is truly a modelless controller which could control a damaged system.

The second experiment exposes some of the limitations of the QIL algorithm in this type of application, though. The experiments conducted with a simple supported beam

model show a less than satisfactory performance improvement. The complexity of the system and the discrete control inputs prevent QIL from ever achieving adequate control. In the beam experiment, the QIL algorithm tries to control a beam under excitation by a disturbance in the first mode. The algorithm learns the disturbance and tries to compensate for it, but unfortunately the control choices available do not exactly cancel the disturbance and consequently, control spillover is the result. The lack of the ability to choose continuously variable controls is the inherent weakness the QIL algorithm suffers in trying to compensate a disturbance or damage. By being forced to select from control inputs designed into the algorithm by the user, the QIL algorithm may be unable to precisely cancel a damaging disturbance or its effects. To alleviate this problem, a heuristic needs to be designed into the algorithm which evaluates each control input for usefulness. If a control input is not needed, then it should be eliminated and another input created to fill the needs of the system.

6.3 Recommendations for Future Work

This work has investigated the viability of QIL as an model-independent active damage controller. Its potential for use in this area of control does invite further study. Some limitations of the algorithm could be overcome by revising the performance index and the forced quantizations. A performance index which weighted the most recent control choices more than early control choices should reduce the premature elimination of certain controls. Incorporating this idea with the a means for the algorithm to refine its

control domain and state space quantizations should eliminate the chatter effect around the origin and possibly allow better disturbance compensation.

Additionally, the algorithm could be made more robust by incorporating or melding other A.I. methods into QIL's basic structure. Specifically, genetic algorithms with their cross-breeding and survival heuristics may allow the algorithm to generate a better initial control. Again this should reduce the premature elimination of certain control inputs. Also, cross-breeding may be a means to allow certain quantizations to be evaluated and either split apart or lumped together accordingly.

Finally, the algorithm should be extended to perform a multiple-input/multiple output control. The goal is to further investigate QIL's ability to handle complex systems. The interactions between multiple control outputs may provide further insight into the mechanism of learning as well. As an added complexity, damage can be introduced into the system to test QIL's ability to adapt its control to a damaged system, once it has learned to control an undamaged one.

REFERENCES

1. Arimoto, S., S. Kawamura, and F. Miyazaki, "Bettering Operation of Robots by Learning," *Journal of Robotic Systems,* Vol **1**(2), pp. 123-140, 1984.

2. Astrom, K. J., "Self Tuning Regulators" Design Principles and Application", Application of Adaptive Control, edited by K. S. Narendra and R. V. Monopoli, pp. 1-68, Academic Press, New York, 1980.

3. Barto, A. G., R. S. Sutton, and C. W. Anderson, "Neuronlike Adaptive Elements that can solve difficult learning control problems". *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC-13, Number 5, pp. 835-846, September/October 1983.

4. Boyce, W.E and DiPrima R.C., Elementary Differential Equations and Boundary Value Problems, John Wiley & Sons, New York, 1986, pp. 456-467.

5. Carbonell, J.G., R. S. Michalski, and T. M. Mitchell, "An Overview of Machine Learning", Machine Learning: An Artificial Intelligence Approach, editors R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Vol. 1, pp. 3-23, Morgan Kaufmann, Palo Alto, 1983.

6. Fagan, G.T. and Robertshaw, H.H., "Discrete Implementation of Positive Position Feedback Analysis and Design Approaches with an Experiment", *Proceedings of AIAA/ASME Adaptive Structures Forum* (AIAA Paper 94-1787), Hilton Head, SC April 21-22, 1994.

7. Glass, B. J., and A. Macalou, "Search-Based Model Identification of Smart-Structure Damage", AD Vol. 24 and Vol 123, Smart Structures and Materials, pp. 33-98, 1991.

8. Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, Reading, 1989.

9. Gomes, A., and M. Silva, "On the Use of Modal Analysis for Crack Identification", *Proceedings of the 8th International Modal Analysis Conference,* Kissimmee, FL., pp. 1108-1115, 1990.

10. Hanagud, S., et. al., "Active Control of Delaminations in Composite Structures", *Proceedings of the 33rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Dallas, TX, pp. 1819-1829, 1989.

11. Kiel, David H., "Active Damage Control using Artificial Intelligence: Initial Studies into Identification and Mitigation", Master's Thesis, Virginia Polytechnic Institute and State University, Mechanical Engineering, 1993.

12. Michie, D., and R. A. Chamber, "BOXES: an experiment in adaptive control", *Machine Intelligence 2*, Oliver and Boyd, Edinburgh, pp. 137-152, 1968.

13. Michie, D., and R. A. Chambers, "Boxes as a model of pattern foundation", *Towards a Theoretical Biology,* Vol. 1., Oliver and Boyd, Edinburgh, pp. 206-215, 1972.

14. Pascoe, J. W., H. H. Robertshaw, and D. H. Kiel, "Inductive Learning Method for Control of Intelligent Structures", *Proceedings of 2nd SPIE Conference,* Orlando, FL., 1994.

15. Rogers, C. A., C. Liang, and S. Li, "Active Damage Control of Hybrid Material Systems using Induced Strain Actuator", *Proceedings of the 32nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference,* Part 2, Washington D. C., pp. 1190-1203, 1991.

16. von Flotow, A. H., and B. Schafer, "Wave-Absorbing Controllers for a Flexible Beam', *Journal of Guidance, Control, and Dynamics,* Vol. 9, Number 6, November-December, pp. 673-680, 1986.

17. Zhang, B. S., and J. M. Edmunds, "A State Learning Controller", *Proceedings American Control Conference*, FM2, pp. 3057-3061, 1992.

VITA

Born in Rochester, New York in 1969, the author was granted his Bachelor's of Science in Mechanical Engineering from the State University of New York at Buffalo in 1991. In the fall of 1991, he began the pursuit of his Master's of Science at Virginia Polytechnic Institute. Currently, he is working in Massachusetts at Zymark, Corp., a small automation company which specializes in robotics and system integration primarily for pharmaceutical, chemical. and biotech companies.